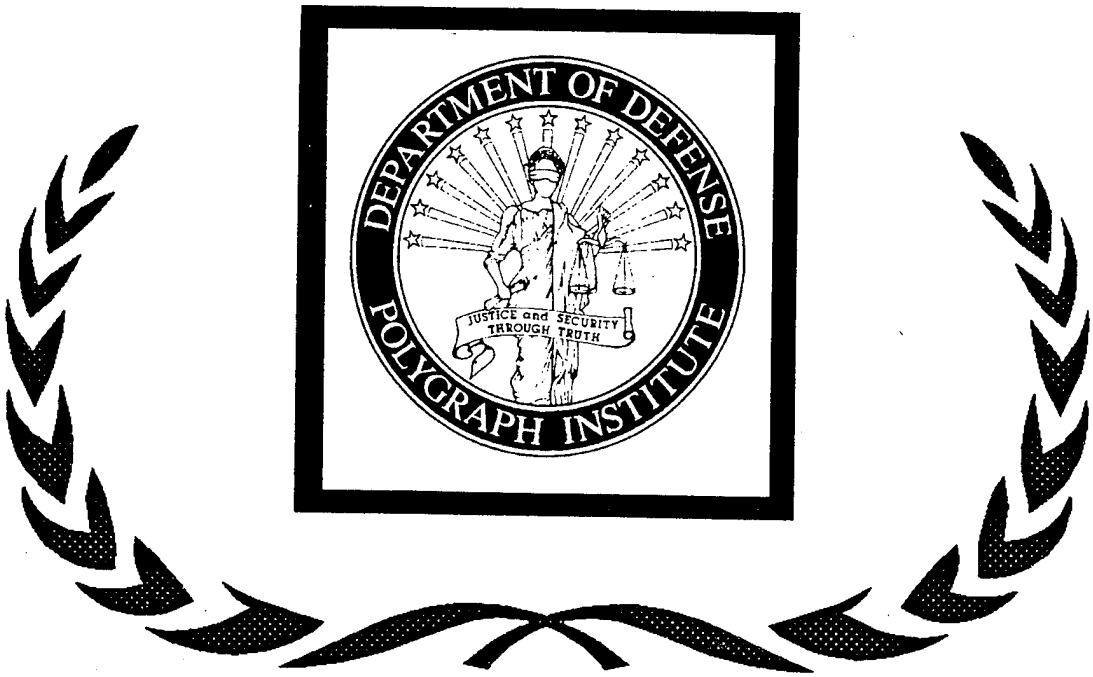


Final Report on the Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph, Volume I of II

R. Benjamin Knapp, Ph.D., Ulka Agarwal, M.S.,
Ramin Djamschidi, M.S., Shahab Layeghi, M.S.,
Mitra Dastamalchi, M.S., and Eric Jacobs, M.S.

December 1995

Department of Defense Polygraph Institute
Fort McClellan, Alabama 36205-5114
Telephone: 205-848-3803
FAX: 205-848-5332

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1995	Final Report Jan 93 - Dec 95

4. TITLE AND SUBTITLE

Final Report on the Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph, Volume I of II

5. FUNDING NUMBERS

DoDPI93-P-0014

6. AUTHOR(S)

R. Benjamin Knapp, Ulka Agarwal, Ramin Djamschidi, Shahab Layeghi, Mitra Dastamalchi, Eric Jacobs

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Department of Electrical Engineering
San Jose State University
P. O. Box 720130
San Jose, California 95172-0130

8. PERFORMING ORGANIZATION REPORT NUMBER

DoDPI96-R-0002

N00014-93-I-0570

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of Defense
Polygraph Institute
Building 3195
Fort McClellan, AL 36205-5114

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

DoDPI93-P-0014

DoDPI96-R-0002

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Public release, distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This project was completed to determine if fuzzy set classification could be used to accurately evaluate data collected during a psychophysiological detection of deception examination. This methodology provides an alternative to the proprietary statistical technique now commonly used. Data collected using both the Modified General Question Technique (MGQT) and the Relevant Only formats were evaluated. An extensive and, arguably, complete set of polygraph data features was identified. These polygraph data features were not individual dependent, examiner dependent, or in any way dependent on apriori or posteriori knowledge (statistics) of the data. A fuzzy K-Nearest Neighbor classifier and an adaptive fuzzy Least Mean Squares classifier were developed. A fuzzy C-Means clustering algorithm which enabled visualization of the data features was also developed. the fuzzy algorithms were "forced" to make a choice of truth versus deception; they could, however, be used to return a number that would, in near real-time, give the examiner an idea of the confidence level of the algorithm. the data were parsed such that 25% of the data were tested using an algorithm developed from the remaining 75% of the data. It is shown that only four features are needed to achieve 100% correct classification of the Relevant Only data and 97% correct classification of the MGQT data. It is suggested that any future research development, or testing or computer classification techniques, including statistical and neural techniques, include the results of this work.

14. SUBJECT TERMS

algorithm, polygraph, deception, truth, fuzzy, fuzzy logic, fuzzy set, psychophysiological detection of deception, computer

15. NUMBER OF PAGES

147

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Report No. DoDPI96-R-0002

Final Report on the Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph, Volume I of II

R. Benjamin Knapp, Ph.D., Ulka Agarwal, M.S.,
Ramin Djamschidi, M.S., Shahab Layeghi, M.S.,
Mitra Dastamalchi, M.S., and Eric Jacobs, M.S.

December 1995

Department of Defense Polygraph Institute
Fort McClellan, Alabama 36205

Director's Foreword

Computer analysis of polygraph charts is one of the key means of improving the accuracy of the polygraph technique. It eliminates variability inherent in human scoring and greatly increases the reliability of the analyses. Computers can analyze factors that are impossible for even the most capable of human examiners to see no matter how thoroughly he or she inspects the polygraph charts. Computers can analyze complex waveforms far faster, in much greater detail, and far more consistently than can humans.

However, it is no easy task to determine the best way to analyze the test data. Many statistical approaches have been used, with varying success. The first major approach used discriminant analysis to differentiate between innocent and guilty subjects. Other avenues being explored include decision trees, logistic regression, and artificial neural networks. All avenues must be explored to find the methods that produce the greatest degree of accuracy.

The approach taken in this study is fuzzy logic using the Fuzzy K-Nearest Neighbor algorithm. Fuzzy logic eschews probability theory, used by most earlier methods, in favor of looking at the degree of membership: whereas, probabilities convey information about relative frequencies (91 out of 100 people with this score are truthful). Fuzzy logic looks at how similar the data are to imprecisely defined properties (this polygraph chart is 91 percent similar to charts from truthful people).

This report consists of four graduate theses by students at San Jose State University working on this program under the guidance of Dr. Benjamin Knapp of the Electrical Engineering Department. They found that fuzzy set classification can evaluate polygraph charts with relatively high accuracy. Care must be taken in generalizing from this result, due to the relatively small number of cases used.

Michael H. Capps
Director

This research was funded through the Department of Defense Polygraph Institute (DoDPI) project DoDPI93-P-0014, under contract number N00014-93-I-0570. The views expressed in this report are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

Abstract

Knapp, B. R., Agarwal, U., Djamschidi, R., Layeghi, S., Dastamalchi, M., & Jacobs, E. Final report on the use of fuzzy set classification for pattern recognition of the polygraph, May 1996, Report No. DoDPI96-R-0002. Department of Defense Polygraph Institute, Ft. McClellan, AL 36205-5114.--This project was completed to determine if fuzzy set classification could be used to accurately evaluate data collected during a psychophysiological detection of deception examination. This methodology provides an alternative to the proprietary statistical technique now commonly used. Data collected using both the Modified General Question Technique (MGQT) and the Relevant Only formats were evaluated. An extensive and, arguably, complete set of polygraph data features was identified. These polygraph data features were not individual dependent, examiner dependent, or in any way dependent on apriori or posteriori knowledge (statistics) of the data. A fuzzy K-Nearest Neighbor classifier and an adaptive fuzzy Least Mean Squares classifier were developed. A fuzzy C-Means clustering algorithm which enabled visualization of the data features was also developed. The fuzzy algorithms were "forced" to make a choice of truth versus deception; they could, however, be used to return a number that would, in near real-time, give the examiner an idea of the confidence level of the algorithm. The data were parsed such that 25% of the data were tested using an algorithm developed from the remaining 75% of the data. It is shown that only four features are needed to achieve 100% correct classification of the Relevant Only data and 97% correct classification of the MGQT data. It is suggested that any future research, development, or testing of computer classification techniques, including statistical and neural techniques, include the results of this work.

Key-words: algorithm, polygraph, deception, truth, fuzzy, fuzzy logic, fuzzy set, psychophysiological detection of deception, computer

Table of Contents

Volume I

Section 1: Time Domain Features for the Fuzzy Set Classification of Polygraph

Section 2: Feature Analysis of the Polygraph

Volume II

Section 3: Pattern Recognition of the Polygraph Using Fuzzy Set Theory

Section 4: Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph

List of Figures

List of Tables

1. Introduction

This is the final report of a 2-year study on the use of fuzzy pattern recognition of polygraph data for the identification of truth versus deception. The goals of this study as stated in the original proposal were to:

1. develop a data parsing algorithm which will process polygraph data obtained from the National Security Agency (NSA) into three domains: time-domain, frequency domain, and correlation domain;

2. design a fuzzy classifier algorithm to accept the featurized data and modify its membership functions based on the error between its classification of the polygraph data and the classification in the NSA files;

3. study the relationship between features (formally called descriptors) and the success of feature classification;

4. study the relationship between the number of membership functions and the success of the data classification and;

5. investigate the feasibility of the classification being performed in a near-real-time scenario.

The data to be used was Modified General Questions Test (MGQT) polygraph data. However, the proposal for the second year of the study introduced the goal of comparing the performance of the developed fuzzy classification system with "zone comparison" polygraph data. Ultimately this was changed to be the simulated "Relevant Only" data obtained from the Department of Defense Polygraph Institute (DoDPI).

This report, including the information described in Sections 1 through 5, shows that all objectives of the original proposal were met. A fuzzy parser and classifier system were developed that could run in near real-time, achieve performance as good or better than the presently available automatic polygraph systems, and identify new features that previously were not used in polygraph classification. Results of 97% correct for the MGQT and 100% correct for the "relevant only" data were achieved. It is shown in this report that while certain features yield good identification across all subjects, a clustering algorithm, fuzzy C-Means, developed in the second phase of this work identified many sets of features that probably should be tried to achieve optimal performance.

2. Phase I: 1993-1994

This first phase of this project developed a complete automatic data parsing system and fuzzy pattern recognition system based on the fuzzy K-Nearest Neighbor (KNN) algorithm. These two elements are summarized below.

2.1 Development of Data Parsing Algorithm

The initial goal of this phase was to be able to read the MGQT data files received from the NSA and separate this data into appropriate features for classification. After consulting with the University of Washington, we were able to develop our own data reading program.

After consultation with experienced polygraph examiners and a detailed review of the polygraph literature, the data reading program was then modified to parse the data into a matrix of features. The feature set included, as outlined in the project proposal, time domain, frequency domain, and correlation domain data. Some examples of the feature set are:

Time Domain Features

- Mean, curvelength, area, and standard deviation for all polygraph channels

- Average of the amplitudes of the peaks in the cardio and respiratory channels

- Derivative of the amplitudes of the peaks of cardio and respiratory channels

- Number of peaks in the cardio and respiratory channels

- Inhalation amplitude/exhalation amplitude of respiratory channels

Frequency Domain Features

- Fundamental frequency of cardio and respiratory signals

- Coherency and cross power spectral density between cardio and respiratory channels

- Power spectral density of cardio and respiratory channels

- Integrated power spectral density for cardio channel

Correlation Domain Features

- Autoregressive parameters (10) for cardio signal

- Cross-correlation between cardio and respiratory channels

In order to classify subjects using the difference between control and relevant responses, and to minimize the size of the feature vector, the features were combined according to the following method: for each feature i (except for the three features corresponding to the cross power spectral density and integrated spectral difference) from each subject j compute:

1. The average control responses $AvgCont_{ij}$
2. The average relevant responses $AvgRel_{ij}$
3. The maximum and minimum control responses $MaxCont_{ij}$ $MinCont_{ij}$
4. The maximum and minimum relevant response $MaxRel_{ij}$ $MinRel_{ij}$

The feature vector components for feature i are then given by the following:

Feature Combination Methods

1. $AvgRel_{ij}$ - $AvgCont_{ij}$
2. $\frac{Avg\ Rel_{ij} - AvgCont_{ij}}{Avg\ Rel_{ij} + AvgCont_{ij}}$
3. $MaxRel_{ij}$ - $MaxCont_{ij}$
4. $MinRel_{ij}$ - $MinCont_{ij}$
5. $MaxRel_{ij}$ - $MinCont_{ij}$
6. $MinRel_{ij}$ - $MaxCont_{ij}$
7. $\frac{MaxRel_{ij}}{MaxCont_{ij}}$

For the three features mentioned previously that cannot be combined as above, each subject j should be computed as:

1. The average of relevant-control responses $Avg(RelCont)_{ij}$
2. The maximum of relevant-control responses $Max(RelCont)_{ij}$
3. The minimum of relevant-control responses $Min(RelCont)_{ij}$

For a complete description of this method, see Volume I, Section 2, Feature Analysis of the Polygraph by M. Dastmalchi.

Ultimately 669 features were automatically extracted from the data. The complete list of all 669 features used in this project are shown in Figure 41 of Volume II, Section 4, Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph. The use of this automatic data parsing algorithm is described in more detailed below in 4.1, Automatic Data Analysis Method.

2.2 Design of Fuzzy Classifier Algorithm

Fuzzy classifier design first focused on the development of a fuzzy set based KNN algorithm. (This work is described in detail in Volume II, Section 3, Pattern Recognition of the Polygraph Using Fuzzy Set Theory, and in Pattern Recognition of the Polygraph Using Fuzzy Classification, Proceedings of the 1994 IEEE International Conference on Fuzzy systems, Vol III, pages 1825-1829.) This algorithm is a supervised learning algorithm which means that training data is presented to the algorithm and then the algorithm is "frozen" and test data is presented. Training on this and all other algorithms in both phases of the study was always performed on 3/4 of the data with testing performed on the remaining 1/4 of the data. The algorithm learned using a set of MGQT data which was divided equally between truthful and deceptive. Since there were 150 deceptive files and only 50 truthful files, the deceptive files were divided into three sets of 50 files each. The relevant only data consisted of 60 non-deceptive and 60 deceptive subjects. A data matrix was created as follows:

x10=[subj#1test1; subj#1test2; subj#1test3; subj#2test1; subj#2test2; subj#2test3,...];

where subj#1test1 consists of the results for the first time subject #1 was asked the entire set of questions, subj#1test2 consists of the results for the second time subject #1 was asked the entire set of questions, etc. This matrix was especially designed to ensure that subject files used for training the algorithm would not coincide with subject files used for testing the algorithm. Thus, the first 3/4 of subjects in the matrix, x10, were used for training, while the remaining 1/4 of subjects were used for testing.

To achieve an accuracy score, the questions were scored individually and then combined at the end of a majority basis. The results of this work are summarized collectively below in 4.2, Classification Accuracy.

3. Phase II: 1994-1995

The second phase of this project dealt with creating an unsupervised clustering algorithm which could identify important features more rapidly, creating another supervised learning algorithm to determine if the fuzzy KNN algorithm was optimal (fuzzy Least Mean Squares; LMS), creating a genetic search algorithm to try to aid in the search for optimal features, and expanding the algorithm testing to look at simulated "Relevant Only" data from DoDPI in addition to the MGQT data. These elements are summarized in the two sections below.

3.1 Comparison of the Fuzzy C-Means, Fuzzy LMS, and Fuzzy KNN Algorithm

An unsupervised clustering algorithm was created to visualize which features allow for larger separation in the truthful and deceptive data clusters. In addition, a supervised learning algorithm, fuzzy LMS, was created to compare with fuzzy C-Means and fuzzy KNN. (This work is described in much more detail in, and partially excerpted from, Volume II, Section 4, Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph, and in Classification of Deception Using Fuzzy Pattern Recognition, Psychophysiology, Volume 31, Supp. 1, August 1994).

The fuzzy LMS system is unique in its application of linguistic knowledge. The use of linguistic knowledge ensures the robustness of the fuzzy system. The use of linguistic information also ameliorates the problem of not having enough reliable numerical data. Unlike classification schemes such as the KNN, the fuzzy LMS algorithm is not entirely dependent on numerical data.

When applied to pattern recognition, fuzzy logic systems can be set up to perform like KNN systems. In KNN systems, numerical data of known class patterns are set up to estimate the probability density distribution of the classes. The probabilities of new data points belonging to the different classes are then computed based on such distribution. Data points around known class samples are then classified into the same class with a higher probability. The fuzzy KNN algorithm modifies the classical KNN algorithm by taking into account the distance between the data point and the known class patterns when estimating the probability. Conceptually this is similar to setting up clusters around all known class samples and calculating the degree of belonging of new data points in the different types of clusters. Other than the exact mathematical equations, that description fits a fuzzy adaptive system where each rule corresponds to a known class pattern and the size of the clusters is the same for all rules.

However, fuzzy adaptive system give up some of the nice theoretical understandings of the KNN systems but gain some practical advantages. The number of rules required are usually much smaller than the number of known samples. Fuzzy logic can usually exploit that to reduce system complexity.

Furthermore, the system complexity for a fuzzy adaptive system stays the same even as new information is available. This is partly a result of the way this algorithm adapts continuously; new information is learned as old information is forgotten. The fuzzy LMS learning technique is like backpropagation, a popular neural network training technique. However, the fuzzy LMS learning algorithm requires few epochs, or iterations, for training. In all our trials the maximum recognition rates for testing data peaked in less than thirty epochs. About 95% of them peaked in less than twenty epochs. This is a few orders of magnitude less than most applications of backpropagation. In many cases the peaks occurred before any training; that is, the system uses only linguistic rules. Here the use of expert knowledge speeds up the training of the system.

The fuzzy C-Means algorithm, unlike fuzzy LMS, is an unsupervised clustering algorithm. Given a set of data, fuzzy C-Means looks for a (usually) predetermined number of clusters within the data points. It does not use any knowledge about the correct, or desired classification of any of the elements. The algorithm only minimizes an objective function, which is the sum of a function of the data points membership values and the distances between the data points and the clusters' centers.

Fuzzy C-Means operates like a black box, given some data, the algorithm automatically computes the results. (Our job is basically to adjust the parameters.) This presents the advantage that different sets of data using different features can be tested in a routine manner. Fuzzy C-Means also presents a way to normalize the different dimensions of the data, just like the use of sigma in the fuzzy LMS algorithm. However, unlike fuzzy LMS, Fuzzy C-Means does not present a method to find the optimal way for such normalization.

The fuzzy LMS algorithm, however, does pose some potential problems of its own. The use of expert knowledge, while a benefit in some senses, may not always be straightforward. For example, in our project we did not have any specific knowledge about polygraphy itself. Whatever we learned, we learned by looking at numerical data. As we tried to find more complicated patterns, patterns involving three, four, or more features, the analysis became more difficult. Naturally, one wishes to automate this process. If we do not reply on some learning procedures, however, rules cannot be automatically found for the fuzzy system. Much research also needs to be done to understand the fuzzy LMS algorithm's learning dynamics. While the same method, gradient descent, is used on both backpropagation and the fuzzy LMS algorithm, the general shapes of the error surface between the two are different. In backpropagation, all parameters have the same range and lie in an uniform neural network structure. In the fuzzy LMS algorithm, the parameters can have different ranges and lie in a fuzzy logic structure which is not completely uniform. The effects of such differences on the shape of the error surface and the learning dynamic are unknown.

A summary of the data comparing these methods is presented in section 4.2 below. All MGQT data was processed as summarized in section 2.2 above.

3.2 Fuzzy C-Means Algorithm on "Relevant Only" Data

The data parsing algorithm was extensively modified to process the "relevant only" data. This data was composed of 166 truthful and 166 deceptive tests with no irrelevant questions asked. Thus the seven feature combination methods described in above, in 2.1, Development of Data Parsing Algorithm, could not be used. Instead, the following four combinations were used:

1. Avg(Feature)

2. Max(Feature)-Min(Feature)

3. Max(Feature)/Min(Feature)

4. Std(Feature)

Also, these files were in an entirely different data format which need to be interpreted for data parsing. (See Volume II, Section 5, Errors in the Relevant Only Data for a summary of incorrect data formats from the "Relevant Only" data.)

4. Summary of Results

The results for the entire project are summarized below. First, the complete automatic data analysis package is summarized including data parsing and classification. Second, a comparison of accuracies amongst the different methods for both MGQT and "relevant only" polygraph data is presented.

4.1 Automatic Data Analysis Method

Below is a description of the automatic data parsing and classification technique developed in this project. Refer to: Volume I, Section 1, Time Domain Features for the Fuzzy Set Classification of Polygraph; Volume I, Section 2, Feature Analysis of the Polygraph; Volume II, Section 3, Pattern Recognition of the Polygraph Using Fuzzy Set Theory; and, Volume II, Section 4, Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph for more complete descriptions.

4.1.1 Parsing the Data

4.1.1.1 Reading the Data

It should be noted that the data reading methods are only important for "off-line" processing and would not be used for near real-time applications.

The data was collected in three phases labeled by the DoD as ERS- 1, ERS-2 and ERS-3. Each polygraph test may consist of one to five charts with each chart consisting of three files. Each chart is a series of questions, usually ten questions. The files are given in DOS file format and must be read and decoded before they can seen.

The following files comprise a chart:

$$EACOWO.011

$$EACOWO.021

$$EACOWO.031

Each of these three files has a specific significance. The .XX3 files are test files which contain the questions which the subjects were asked. The .XX1 and .XX2 files are encoded in a specific format created by Axciton polygraph testing devices. These files can be decoded by a program entitled read3. Read3 can be invoked in DOS as in the following example:

read3 $$EACOWO.011 output1

read3 $$EACOWO.021 output2

read3 $$EACOWO.031 output 3

The read3 command decodes the data in files *.011, *.021 and *.031 and writes them in ASCII files entitle output1, output2 and output3, respectively. Output2 and output3 contain the actual signals from four polygraph channels with a timing signal which shows the times when the questions were asked. The output files were labeled such that minimal confusion was allowed. For example, the output file for non-deceptive subject 45, text file .XX3 compiled during phase ERS-1 reads:

nd45t3.ex1

4.1.1.2 Feature Extraction

After the polygraph files are decoded and written into output files, they can be processed in MATLAB. MATLAB is a commercially available mathematical analysis program which runs on a PC, Macintosh, and most UNIX platforms. The feature extraction process consists of a MATLAB program which extracts features for all files and saves them in a matrix consisting of subjects and features. The main feature extraction program is a MATLAB routine called Do.M. This program extracts the pre-selected 52 features, from each subject, contained in the variable feature_list. Feature_list is a MATLAB matrix which includes the names of the feature extraction routines. In each row of the feature_list matrix, a feature extraction routine is named along with the channel number(s) this routine will be applied to. The mean, standard deviation, maximum subtracted from the minimum and the maximum divided by the minimum is taken from the extracted features. These four results are put into a matrix which is then put into a larger matrix called x10.mat, consisting of all non-deceptive and deceptive subjects and all 52 features from the feature list.

4.1.2 Classifying the Data

After the data is parsed in DOS and MATLAB, the classifying process takes place entirely in MATLAB.

4.1.2.1 K-Nearest Neighbor Algorithm

The main program which runs the KNN algorithm is called fknn which is written in the C programming language. The file interacts with MATLAB by reading and writing files in MATLAB format, that is .mat files. This algorithm is implemented by the program fknn which opens a MATLAB data file, reads the training matrix, classifies each entry in the testing matrix and writes the result in an output file. The file from which this program receives information from is "fdatafile.mat" which is in MATLAB file format.

Because the KNN algorithm has been automated, it can be run in only a few simple steps. For a complete description of this process see Volume II, Section 3, Pattern Recognition of the Polygraph Using Fuzzy Set Theory. Before running the algorithm a few variable must be determined. For example, for the "relevant only" data:

1. A single variable "C," the number of classes was set equal to two; one for deceptive and one for non-deceptive.

2. A single variable "K," determines how many different points surrounding a chosen point will be compared to it and classified. The parameter "K" in the KNN algorithm was varied from one to ten throughout the simulations.

3. A single variable "M," the coefficient in the fuzzy algorithm was set equal to two.

4. A training matrix "P," contains a set of feature vectors. Each vector is a column of the matrix. There were fifty deceptive and fifty non-deceptive tests used for training. The combination of features to be tested is also entered in this matrix.

5. A class membership matrix "T," contains the membership values of the training set vectors to the classes. This matrix was set that a one was displayed for a non-deceptive detection and a zero for a deceptive detection.

6. An input matrix "U," which contains a set of unclassified feature vectors contained the rest of the tests not used for training. These remaining tests make up the testing matrix. The same combination of features entered in "P" are to be entered in the "U" matrix.

7. Threshold, which is varied from 0.2 to 0.8 throughout the simulations.

Once the matrix X10.mat is loaded in MATLAB, the KNN algorithm can be invoked by simply trying "KNN." The user will then be asked to enter a numerical value for the K parameter in the KNN algorithm. Parameters chosen between one and ten have been found to produce the best results. Once the "K" parameter has been entered, the number of correct deceptive and non-deceptive identifications can be obtained by entering the following:

sum(fresult(1,1:116)>0.5) non-deceptive

sum(fresult(1,117:232)<0.5) deceptive

The correct detection for non-deceptive data is shown by a one, so the threshold is greater than 0.5. The percent correct for the deceptive data can be obtained by dividing the number of correct deceptive classifications by 166. This same process works for the non-deceptive data. Finally, the total correct detection percent is obtained by taking the average of the two percentages.

4.1.2.2 Fuzzy C-Means

The fuzzy C-Means algorithm for MGQT data has been made user friendly through automated push buttons written in MATLAB (see Figure 1). These buttons allow the user to execute the feature extraction and classification process without an understanding of the complexity of each program used in the algorithm. With minor modifications, the push buttons can be used for the "relevant only" data as well.

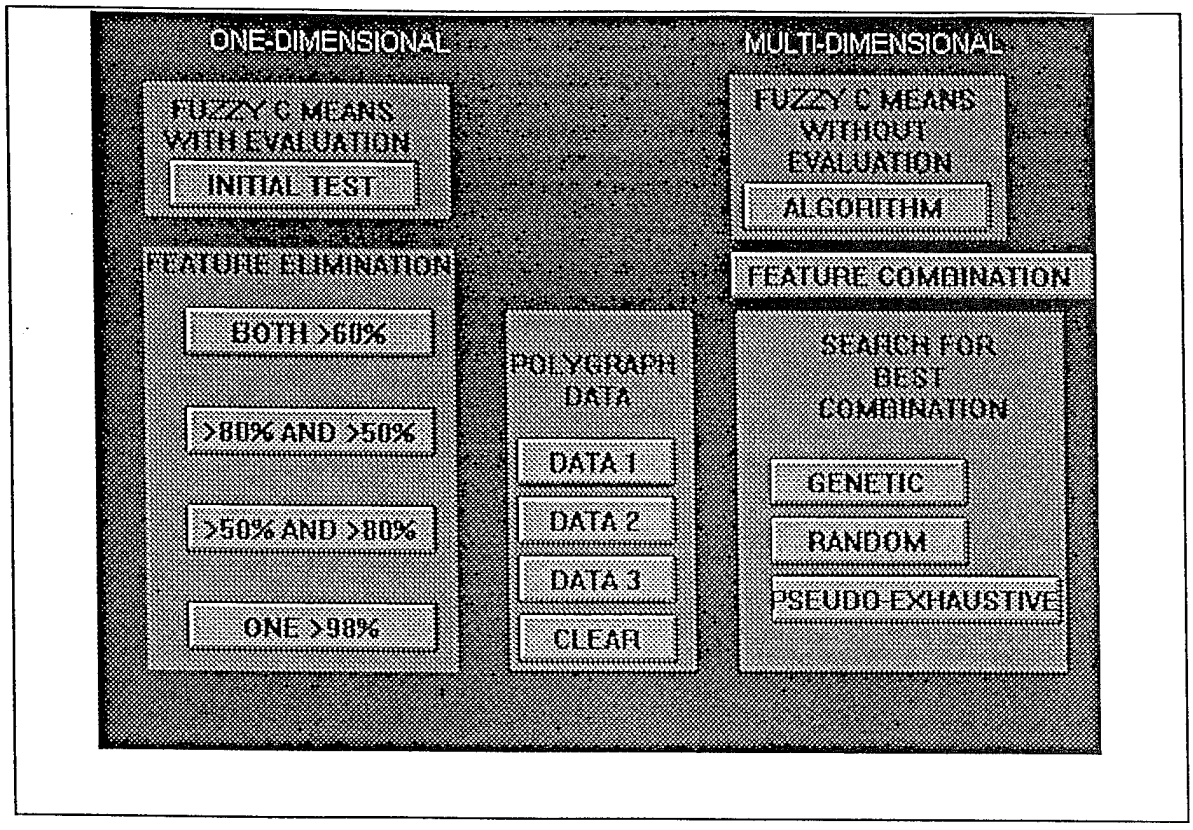


Figure 1: User Interface for Fuzzy C-Means Clustering Algorithm

Before running the algorithm a few variables must be determined. For example, for the "relevant only" data:

1. The "temp" matrix in the fc_means program was set equal to the dimensions (1,332).

2. The threshold was varied from 0.2 to 0.8 for each different simulation that was run.

3. Combination of features to be tested can be changed as described below.

The following execution process is necessary only if the push button automation is not used. After the matrix X10.mat is loaded, the user must type the following to run the algorithm:

[Uik,z] = fc_means(5,0.000005,x10([8 23 24],:))

The z parameter is the number of iterations made by the algorithm to obtain the results and Uik is the membership values. To calculate the correct detection of non-deceptive and deceptive subjects, the user must type the following:

sum(Uik(1,1:166)<0.5) non-deceptive

sum(Uik(1,117:332)>0.5) deceptive

where 0.5 is the selected threshold for this particular simulation. The percent correct for each class can be determined by dividing the number correct by the total number. The total percent correct is then obtained by averaging the two percentages.

4.1.2.3 Least Mean Squares Algorithm

The LMS fuzzy adaptive filter is a nonlinear adaptive filter which makes use of both linguistic and numerical information concerning the physical characteristics of the polygraph data in their natural form. This filter is constructed from a set of changeable fuzzy IF-THEN rules. We have the choice of setting the rules according to our experiences and incorporating them directly into the filter, or initializing the rules arbitrarily. Before running the algorithm a few variables must be determined. For example, for the "relevant only" data:

1. The number of training subjects was set equal to 100.

2. The "running time," how often the algorithm goes through the data, was set to 70.

3. Different combinations of the features were changed manually for each different simulation.

After the matrix X10.mat is loaded the user must simply type:

lmstest

The total percent correct of deceptive and non-deceptive data is automatically displayed under the variable "maximum."

4.2 Classification Accuracy

4.2.1 MGQT

Table 1 shows a comparison of the best results for each of the classification algorithms found in this study. (See Volume II, Section 4, Use of Fuzzy Set Classification for Pattern Recognition of the Polygraph for a more complete description of how this comparison was performed.) It should be noted that the optimum features found for the fuzzy C-Means and the fuzzy KNN algorithms were different. This is important because it means that if both algorithms were run on a given subject, their results could be independent and corroboratory. The fuzzy LMS algorithm was simply run using the optimal four features found for the fuzzy C-Means algorithm. The method number refers to the seven feature-combination methods described in section 2.1 above. The three data sets refer to the fact that the 150 deceptive files were separated into three files of 50 and compared to the 50 non- deceptive files.

Table 1.
Classification Technique Accuracies (% correct) using GQT Data[a]

Featured Used	Method[b]	Data Set 1	2	3
	Fuzzy C-Means			
Ampl of Peaks (High Freq Cardio)	4	93	87	97
Max-Min (High Freq Cardio)	7			
Std (GSR)	2			
Std (GSR)	4			
	Fuzzy KNN			
Max (GSR)	1	86	80	91
Max (Lower Resp)	6			
Max (Upper Resp)	3			
Max-Min (High Freq Cardio)	4			
	Fuzzy LMS			
Ampl of Peaks (High Freq Cardio)	4	81	83	83
Max-Min (High Freq Cardio)	7			
Std (GSR)	2			
Std (GSR)	4			

[a] These results are based on comparisons of 15 truthful and 15 deceptive files for each data set as described above in 4.2.1, MGQT. [b] The method refers to the feature combination methods described above in 2.1, Development of data parsing algorithm.

4.2.2 "Relevant Only"

For the "relevant only" data the fuzzy C-Means algorithm was used since it achieved the best performance for the MGQT data. Table 2 shows the summary of results for different combinations of the four optimal features described in Table 1 above. The different columns represent the different fuzzy thresholds (see Volume II, Section 4, Use of fuzzy set classification for pattern recognition of the polygraph for details). Note that for one of the combination of three features a score of 100% correct for both deceptive and non-deceptive was achieved.

Table 2.
Classification Accuracies (rounded % correct) of "Relevant Only" Data using Fuzzy C-Means Algorithm and Different Feature Combinations[a]

Features	Method[b]	Status[c]	Fuzzy Threshold Truth/ Deception Boundary Value[d] 0.2	0.3	0.4	0.5	0.6	0.7	0.8
1									
Std(GSR)	2	N	100	96	78	49	11	1	3
Std(GSR)	4	D	0	3	28	58	93	99	96
Ampl of Peaks[e]	4								
2									
Std(GSR)	2	N	48	45	36	30	29	24	34
Std(GSR)	4	D	24	30	32	36	45	48	54
Max-Min[e]	7								
3									
Std(GSR)	2	N	48	0	5	33	77	99	100
Ampl of Peaks[e]	4	D	24	1	5	32	71	99	100
Max-Min[e]	7								
4									
Std(GSR)	4	N	48	45	34	3	22	23	4
Ampl of Peaks[e]	4	D	24	30	6	65	54	5	66
Max-Min[e]	7								
5									
Std(GSR)	2	N	100	99	95	67	23	1	54
Std(GSR)	4	D	0	0	5	33	71	99	33
Ampl of Peaks[e]	4								
Max-Min[e]	7								

[a] These values are based on comparisons of 15 non-deceptive and 15 deceptive files.
[b] The method refers to the feature combination methods described above in 2.1, Development of data parsing algorithm. [c] Status indicates that subjects were deceptive (D) or non-deceptive (N). [d] Each column represents a different fuzzy threshold value for the truth/deception boundary (see Volume II, Section 4, Use of fuzzy set classification for pattern recognition of the polygraph for details).
[e] High frequency cardio.

5. Conclusions

This project achieved all goals set in the phase 1 and 2 proposals:

1. A data parsing algorithm was developed which will process polygraph data obtained from the NSA into three domains: time- domain, frequency domain, and correlation domain;

Summary: Over 90 features were extracted from the polygraph data. For the MGQT data, seven methods of combination were used; thus achieving over 650 features. Some of these features were chosen from careful interviewing of several polygraph examiners. Other features were chosen from heuristic examination of the data. Still others were chosen simply to represent all aspects of the waveforms. None of these features were chosen based on the statistics of the input signal. It was hypothesized and subsequently shown that "guessing" about the statistics of the signals was not necessary for accurate classification of truth versus deception.

The results of goal #1 are important to the future of polygraph examination for two primary reasons. First, an extensive and arguably complete set of features of the polygraph data was identified. Second, the features of the polygraph are not individual dependent, examiner dependent, or in any way dependent on apriori or posteriori knowledge (statistics) of the data.

2. two fuzzy classifier algorithms were designed to accept the featurized data and modify the algorithm's membership functions based on the error between the algorithm's classification of the polygraph data and the classification in the NSA files;

Summary: A fuzzy KNN classifier and an adaptive fuzzy (fuzzy LMS) classifier were developed. While it was hypothesized that the fuzzy classification of the polygraph would achieve results comparable to statistical techniques, this was not known before the project began. It can now be said that these fuzzy algorithms can achieve comparable, if not superior results, to statistical techniques.

The results of goal #2 are important to polygraph examination for two primary reasons. First, an alternative method to the proprietary statistical technique now commonly used will give examiners a corroborating data analysis algorithm that can be modified as needed. Second, the fuzzy algorithms were "forced" to make a choice of truth versus deception. They could, however, be used to return a number that would, in near real-time, give the examiner an idea of the confidence level of the algorithm. This would enable the examiner to target further questioning to the areas where the algorithm is yielding ambiguous results.

3. a clustering algorithm was developed to examine the relationship between the features chosen and the success of the classification;

Summary: A fuzzy C-Means algorithm was developed which enabled visualization of the

features of the polygraph data. Using this algorithm, several features were found to be superior to all others. (These features were superior for the data we tested. We would need to analyze many more polygraph charts before this result could become definitive.)

The results of goal #3 are important for two primary reasons. First, an algorithm was developed to allow researchers to investigate features and determine which one (or combination of up to four) classify the data the best. Second, several features were found to be superior to all others. These features are:

- the amplitude of the peaks and the dynamic range of the cardiograph signal,
- the standard deviation and the maximum value of the galvanic skin response, and,
- the maximum value of the lower and upper respiratory signal.

4. relationships were found between the number of membership functions and the success of the data classification up to four simultaneous features;

Summary: It was not clear before the project began how many features or membership functions would be needed to correctly classify the polygraph data. It was shown that only four features were needed to achieve 100% correct classification of simulated relevant only data and 97% correct classification of actual MGQT data.

The results of goal #4 are extremely significant. An algorithm that needs very few features to achieve such high classification can be made to execute in a near real-time environment. These features can also be tested easily by other classification systems.

5. the feasibility of the classification being performed in a near-real-time scenario was shown.

Summary: As mentioned previously, a fuzzy algorithm executing in near real-time classified the polygraph data at accuracy levels as high or higher than have ever been achieved before. These results were achieved by analyzing polygraph subject files that were previously "unseen" by the algorithm. They were also achieved without allowing for any "don't know" results.

The positive results of goal #5 and all the goals mentioned previously means that there exists a new technique for automated classification of polygraph data. Any further research, development, or testing of computer classification techniques, including statistical and neural techniques, must include the results of this work. If not, an important, highly accurate, and possibly superior classification method will have been ignored.

Report No. DoDPI96-R-0002

Time Domain Features for the Fuzzy Set Classification of Polygraph Data

Eric Jacobs
San Jose State University
Department of Electrical Engineering
San Jose, CA 95106

November 1993

Department of Defense Polygraph Institute
Fort McClellan, AL 36205

Table of Contents

List of Tables

List of Figures

1.1 History

The first attempt to use a scientific instrument in an effort to detect deception occurred around 1895 [3]. That was the year that Cesar Lombroso published the results of his experiments in which a hydrosphygmograph was used to measure the blood pressure-pulse changes of criminals in order to determine whether or not they were deceptive. Although the hydrosphygmograph was originally intended to be used for medical purposes, Lombroso found that it worked well for lie detection. Lombroso may have been the first to use a peak of tension test format. This was done by showing a suspect a series of photographs of children, one being the victim of sexual assault. If the suspect did not react more to the victims picture than the pictures of the other children, Lombroso concluded that the suspect did not know what the victim looked like and therefore was not the alleged perpetrator.

In 1914 Vittorio Benussi published his research on predicting deception by measuring recorded respiration tracings [4]. He found that if the length of inspiration were divide by the length of expiration, the ratio would be larger after lying than before lying and also before telling the truth than after telling the truth. In 1921 John A. Larson constructed an instrument capable of simultaneously recording blood pressure pulse and respiration during an examination [3][4]. Larson reported accurate results which prompted Leonarde Keeler to construct a better version of this instrument in 1926 [3][4].

The use of galvanic skin response in lie detection began during the turn of the century. It's usefulness, however, did not become evident until the 1930's during which time several articles written by Father Walter G. Summers of Fordham University in New York [4]. In these articles he reports over 90 criminal cases in which examination using the galvanic skin response had all been successful and confirmed by confession or supplementary evidence. The usefulness of the galvanic skin response prompted Keeler to add an galvanometer to his polygraph. At the time of Keelers death in 1949, the Keeler Polygraph recorded blood pressure-pulse, respiration, and galvanic skin response [3].

1.2 Modern Test Formats

The effectiveness of a polygraph examination is often the result of the test format that is used. A polygraph test format consists of an ordered combination of relevant questions about an issue, control questions that provide a physical response for comparison, and irrelevant questions that also provide a response or the lack of a response for comparison [1][4]. Three general types of test formats are in use today. These are Control Question Tests, Relevant-Irrelevant Tests, and Concealed Knowledge Tests. Each of the general test formats may have a number of more specific variations. Each test consists of two to five charts containing a prescribed series of questions. The test format that is used in an examination is determined by the test objective [3][4].

The concealed knowledge test, also called peak of tension test, is used when facts about a crime are known only by the investigators and not by the public. In this case, a subject would not know the facts unless he or she was guilty of the crime. For example, if a gun was used in a crime and the public did not know the caliber, an examiner could ask a suspect if it was a 22 caliber , a 38 caliber, or a 9mm. If the gun used was a 9mm

and the suspect was deceptive, a polygraph chart would probably indicate evidence of deception.

A control question test is often used in criminal investigations. In this type of test a series of relevant, irrelevant, and control questions are asked. A relevant question is one which is specific to the crime being investigated. For example, " Did you molest the child?". A control question is designed to make the subject feel uncomfortable. It is not specific to the crime being investigated however it may be related in an indirect way. A control question that could follow the relevant question stated above is "Have you ever forced yourself on another person sexually ?". The control questions are compared to the relevant questions and if the responses to the relevant questions are greater, the subject is usually classified as deceptive. Irrelevant questions are used as buffers. Examples of irrelevant questions are "Are the lights in this room on?" or "Is today Monday?".

Relevant-Irrelevant tests are usually used to test people trying to obtain security clearance or get a job. In this test, relevant questions are compared to irrelevant questions. Very few control questions are asked. The purpose of control questions in this test is to make sure that the subject is capable of reacting at all.

1.3 Present Day Equipment

The most popular polygraph machines today are the Reid Polygraph developed in 1945 and the Axciton Systems computerized polygraph developed in 1989 [1][11]. The Reid polygraph scrolls a piece of paper under pens that record the biological signals. The Axciton polygraph digitizes physiological signals and uses a computer to process them. The sampling frequency of the Axciton machine is 30 Hz. Axciton provides a computer based system for ranking the subject responses but allows printouts of the charts to be scored by hand the traditional way. The Axciton and Reid polygraphs are shown in figures 1 and 2 respectively.

Both machines record the same biological signals using standard methods. Blood pressure is measured by placing a standard blood pressure cuff on the arm over the brachial artery. Respiration is monitored by placing rubber tubes around the abdominal area and the chest of the subject. This results in two signals, an upper and lower respiratory signal. Skin conductivity is measured by placing electrodes on two fingers of the same hand.

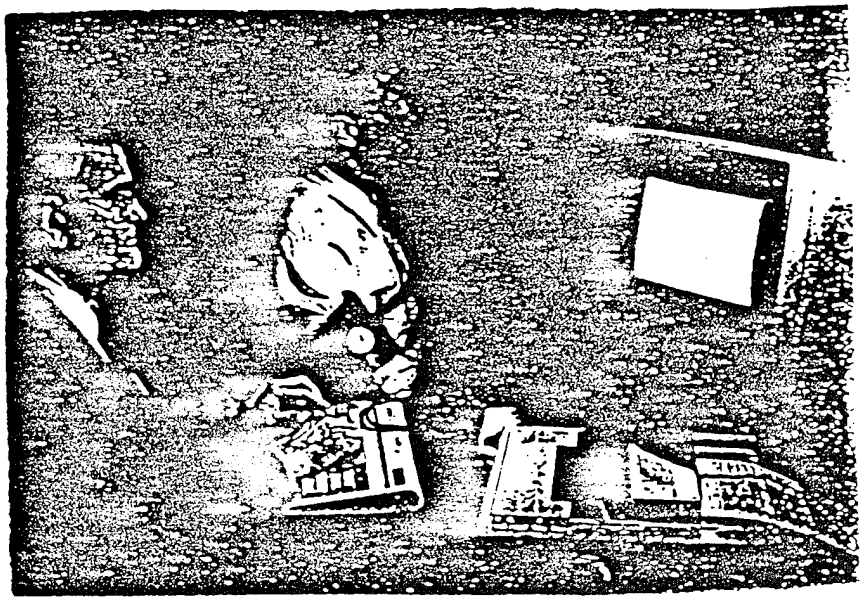


Figure 1 Axciton Polygraph [1]

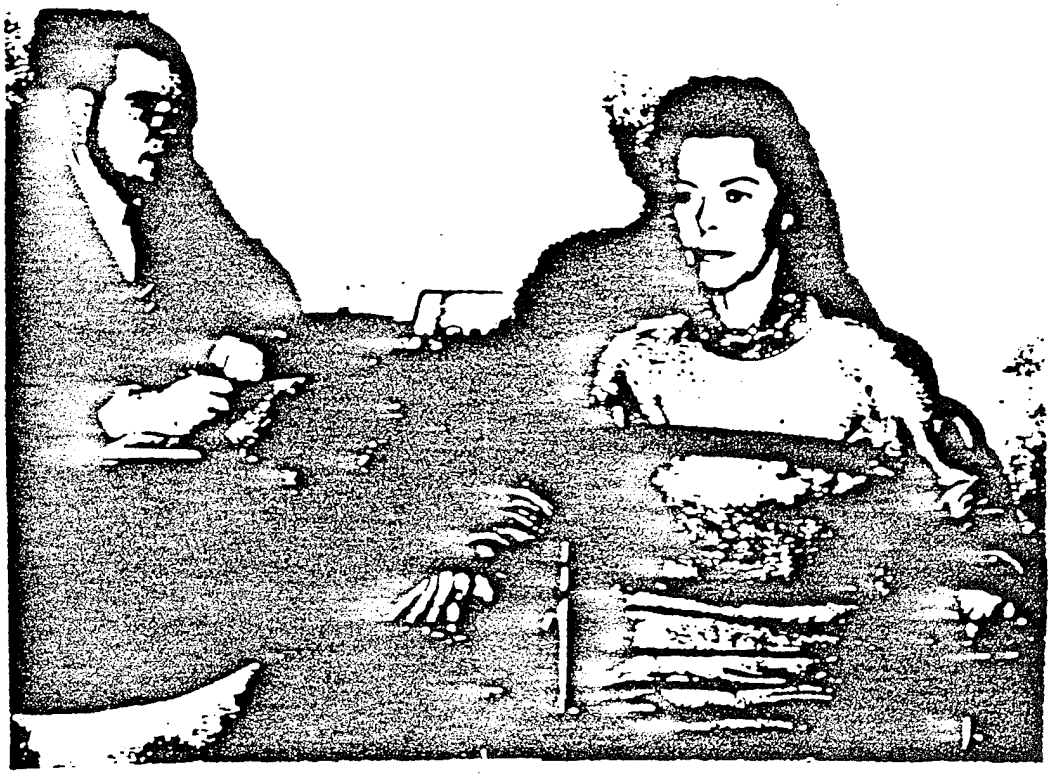


Figure 2 Reid Polygraph [3]

2.1 Fuzzy Set Theory

In 1965 fuzzy sets were introduced by Lofti Zadeh [5][6]. They provided a new way to represent vagueness and made description of many situations much easier. For example, it is not practical to say that all temperatures below 72 degrees Farenheit are cold and all temperatures above are hot. Instead, temperatures between 50 and 72 would by described as cool, temperatures between 30 and 50 would be considered cold, and anything below 30 would be very cold. One way to describe this situation is through the use of fuzzy set theory. In fuzzy set theory an element is not defined as belonging or not belonging to a given set. Instead, it has a degree of membership in a set which is characterized by a compatibility function uA [6] [7]. The compatibility function, also called a membership function, states the degree of membership in a set "A" and has a range [0,1]. An illustration of how this applies to the temperature example above is illustrated in figure 1 and described below.

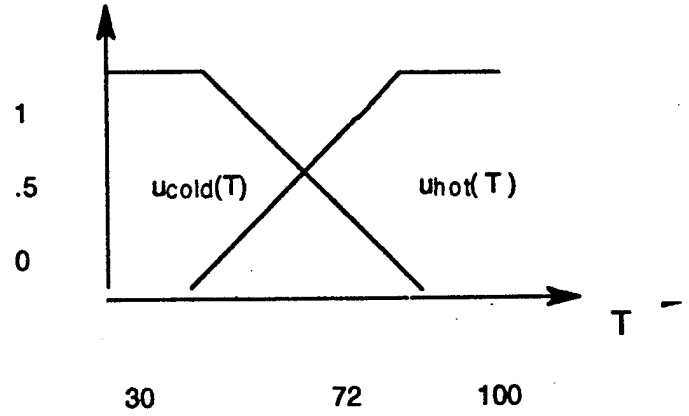


Figure 3 Compatibility functions ucold(T) and uhot(T) vs. temperature.

Here, $u_{cold}(T)$ and $u_{hot}(T)$ are the degrees of membership in each set and T is the temperature in Farenheit. Figure 1 shows that the temperatures around 72 degrees have membership in $u_{cold}(T)$ and $u_{hot}(T)$. These memberships have values around .5 which represents cool or warm. As the cooler temperatures decrease, $u_{cold}(T)$ increases thus representing a colder situation. Once the temperatures become less than 30 degrees, $u_{cold}(T)$ obtains a membership value of 1 which indicates very cold temperatures.

Fuzzy set theory is often thought of as another form of probability theory. In actuality, the two are very different [8]. In Bayesian probability theory, elements either belong or do not belong to a given set, and a probability density function determines the likelihood. For example, a light may be either on or off and the probability of either event occurring will depend on some statistical parameters (Is the room occupied? Is it dark out? etc.). The following is an example of the difference between fuzzy logic and Bayesian probability theory [6].

Example 1

Let L = set of all liquids, and let fuzzy subset l = {all (potable) liquids}. Suppose you had been in the desert for a week without drink and you came upon two bottles marked "C" and "A" as in figure 4a.

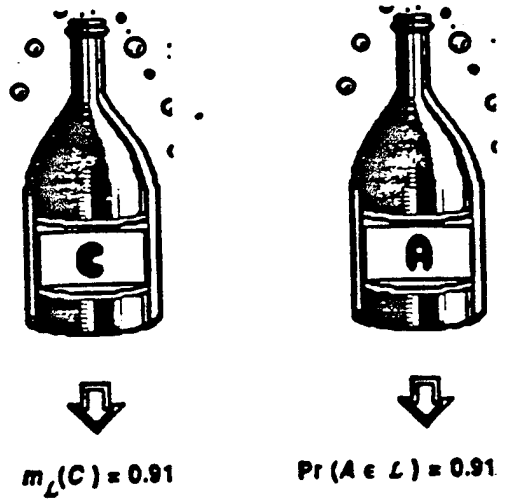


Figure 4a Liquids before observation

Confronted with this pair of bottles, and given that you must drink from the one that you choose, which would you choose to drink from? Most readers, when presented with this experiment, immediately see that while "C" could contain, say, swamp water, it would not (discounting the possibility of a Machiavellian fuzzy modeler) contain liquids such as hydrochloric acid. That is, membership of 0.91 means that the contents of "C" are fairly similar to perfectly potable liquids (e.g., pure water). On the other hand, the probability that " A" is potable = 0.91 means that over a long run of experiments, the contents of A are expected to be potable in about 91% of the trials; in the other 9% the contents will be deadly - about 1 chance in 10. Thus, most subjects will opt for a chance to drink swamp water.

There is another facet to this example, and it concerns the idea of observationion. Continuing then, suppose that we examine the contents of "C" and "A" and discover them to be as shown in figure 4b. Note that, after observation, the membership value for "C" is unchanged while the probability value for A drops from 0.91 to 0.0.

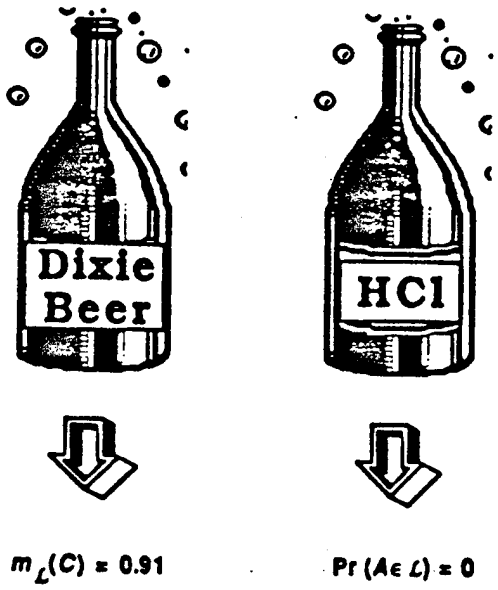


Figure 4b Liquids after observation

This example shows that these two models possess philosophically different kinds of information: fuzzy memberships, which represent similarities of objects to imprecisely defined properties; and probabilities, which convey information about relative frequencies.

3.1 MGQT

The test format used in this project was the MGQT test format. It is a type of control question test in which relevant, irrelevant, and control questions are asked in the order given in table 1 [9][12]. Before each test, the questions that will be asked are discussed with the subject. The series of questions is asked three times in the order specified in table 1. This produces three test charts. The examiner waits about 20 seconds between each question.

Not all of the Axciton charts used in this study follow the format of table 1 exactly. Many examiners rearranged the order in which the questions were asked. All polygraph charts used, however, were variations of this test. For example, one examiner used a test format in which questions 3 and 4 were switched. Many of the examiners changed the order in which the questions were asked in the second and third charts.

Question	Type of Question
1	irrelevant
2	irrelevant
3	relevant
4	irrelevant
5	relevant
6	control
7	irrelevant
8	relevant
9	relevant
10	control

Table 1 MGQT question format

4.1 File Formats

Axciton files, digitized polygraph data from the axciton polygraph, were obtained from the National Security Agency (NSA) in standard MSDOS format. The sampling frequency of the data was 30Hz. Each test consisted of nine files. The labling of the files is shown in table 2 and the purpose of each file is explained below.

Chart 1	Chart 2	Chart 3
$$xxxxxx.011	$$xxxxxx.021	$$xxxxxx.031
$$xxxxxx.012	$$xxxxxx.022	$$xxxxxx.032
$$xxxxxx.013	$$xxxxxx.023	$$xxxxxx.033

Table 2 File format

As stated in the section above, each examination is composed of three charts. The chart number is specified by the second number after the period. The third number after the period represents the type of file.

$$xxxxxx.0x1 is the event marker file which contains the length of the chart and the event markers. The start and end of an examiners question is marked with a 0 and 1, respectively. The beginning of the subjects response is indicated with a 2 and the rest of the file is marked with 9's. File $$xxxxxx.0x2 is the file containing the biological signals. These signals correspond to the marker file. File $$xxxxxx.0x3 contains the questions and labels them relevant, irrelevant, or control.

An ASCII file of five columns is created by using $$xxxxxx.0x1 and $$xxxxxx.0x2 and a program provided by the NSA. An example of this file along with a description of the function of each file is shown in table 3 [12].

	Event Marker	**FileChart Data**	**FileQuestion TextFile**
	$$xxxxxx.0x1	$$xxxxxx.0x2	$$xxxxxx.0x3
Axciton File	Contains the length of the chart, the number of channels, and the position of the event marker.	Contains the digitized series values formatted according to flags in the Event Marker File.	Contains the script of of questions or a shorthand script of questions.
Processing Notes	Becomes the 5th column of ASCII file. 0=start of a question 1=end of a question 2=start of response 9=No Event Marker	Becomes 1st-4th columns of ASCII file. Column 1-GSR Column 2-Cardio Column 3-Upper Resp Column 4-Lower Resp	Files used to determine deviations from standard test format.

ASCII File Format (with column labels)

	File Row	GSR	Cardio	UR	LR	EvMark
DOS File	1	1983	1931	1482	1083	9
	2	1983	1922	1483	1084	9
	3	1983	1913	1483	1084	9
	4	1983	1906	1483	1085	9

Table 3 File description and example

5.1 Preprocessing

MATLAB was used to display the signals and implement all of the filters and feature extraction algorithms. First, the four biological signals were processed into six channels. Hamming windowed FIR filters were used to create these channels and eliminate noise. A low frequency cardiovascular channel was produced by lowpass filtering the cardiovascular signal at .5 Hz using a 134 tap lowpass filter. Then, a high frequency cardiovascular channel was produced by highpass filtering the cardiovascular signal at .5 Hz using a 134 tap highpass filter. The derivative of the low frequency channel was then used to create a third channel. To eliminate noise, the upper and lower respiratory signals were lowpass filtered at 1.2 Hz using a 160 tap filter. Noise was eliminated from the galvanic skin response by using a 100 tap lowpass filter with a cutoff frequency of .5 Hz. Any DC trends that existed within a chart were eliminated using the detrend function in MATLAB. This function finds the best straight line fit to the data and then subtracts the line from the data. Each signal was normalized by dividing by its standard deviation. The raw data and results of this processing are shown in figures 5-14.

Fragments of each signal were accessed before features were extracted. These fragments were successfully used by Brian M. Duston of the Naval Control and Ocean Surveillance Center in his study and are given in table 4 [9]. The start and end points given in table 4 refer to the time elapsed after the question was asked by the examiner.

Channel	Start	End
GSR	2 sec.	14 sec.
Upper respiratory	2 sec.	18 sec.
Lower respiratory	2 sec.	18 sec.
Low frequency cardiovascular	2 sec.	18 sec.
High frequency cardiovascular	3 sec.	9 sec.
Derivative of low frequency cardiovascular	0 sec.	8 sec.

Table 4 Time fragments used in feature extraction

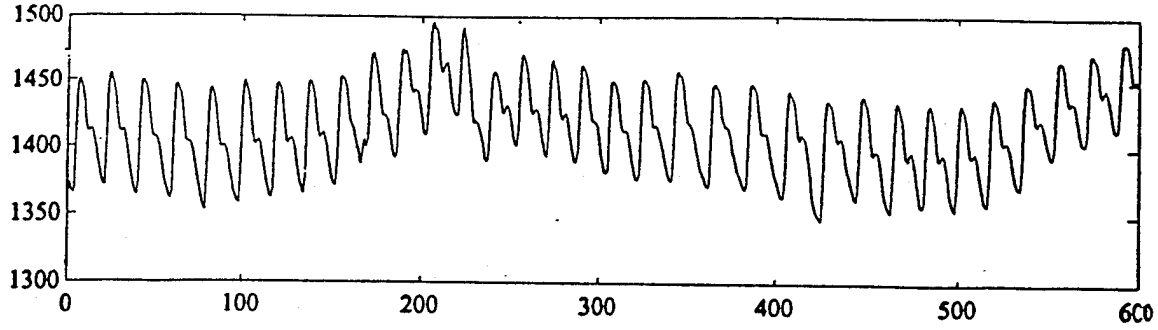


Figure 5 Cardiovascular

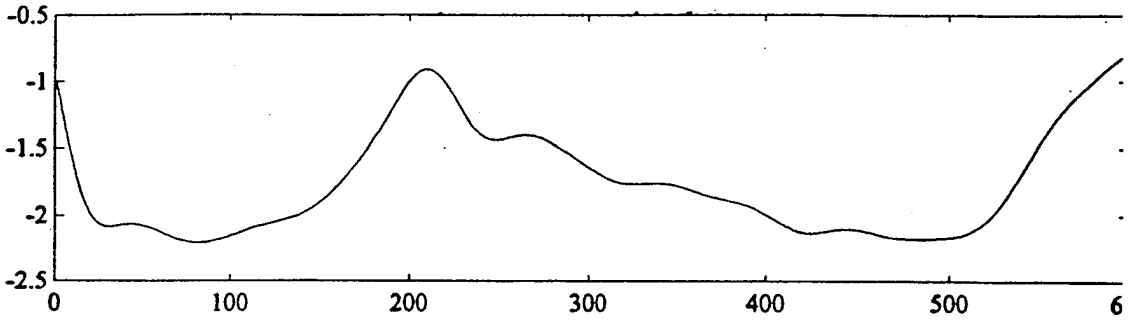


Figure 6 Preprocessed Low Frequency Cardiovascular

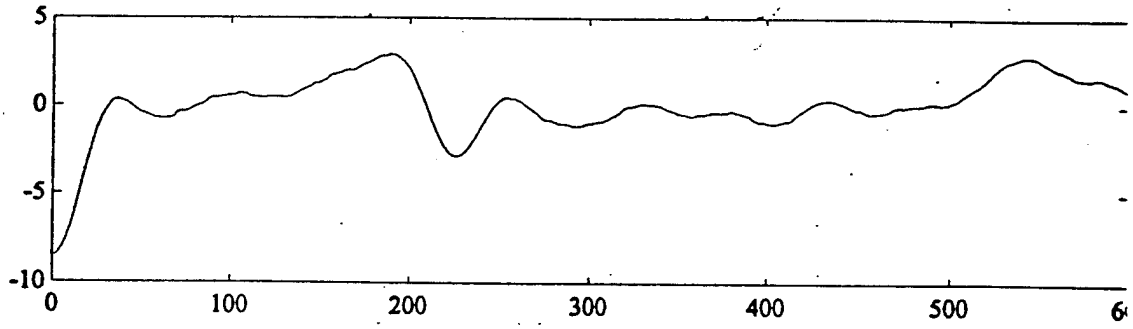


Figure 7 Preprocessed Derivative of Low Frequency Cardiovascular

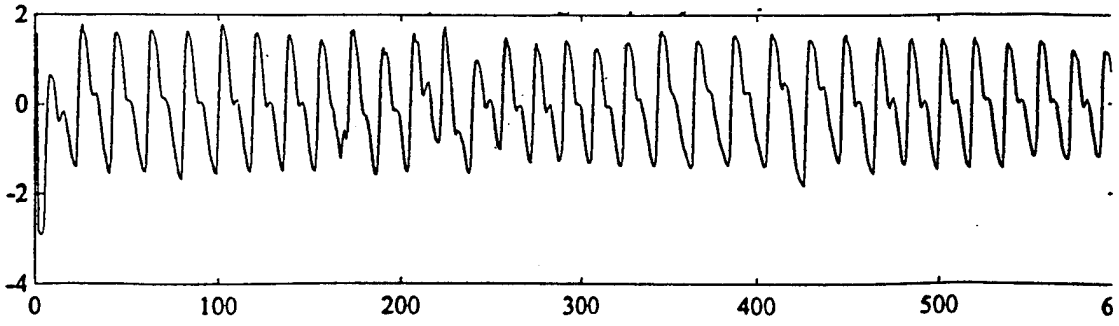


Figure 8 Preprocessed High Frequency Cardiovascular

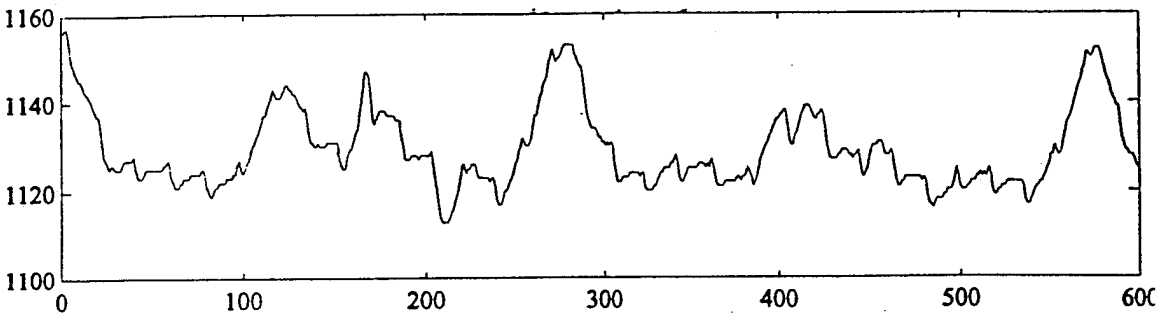


Figure 9 Upper Respiratory

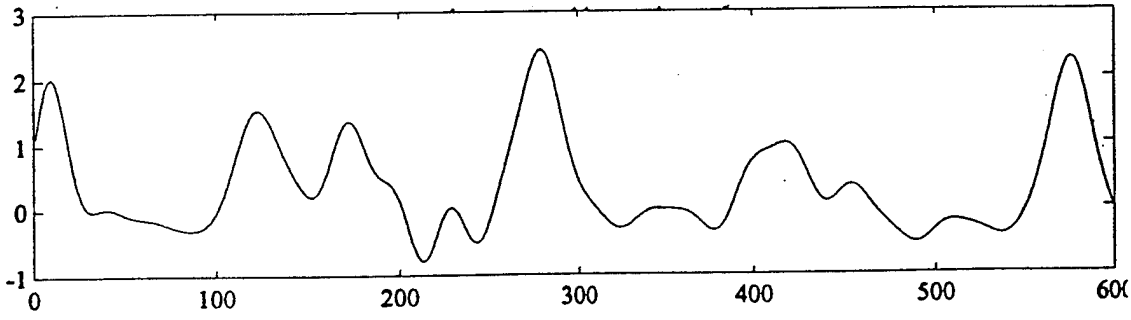


Figure 10 Preprocessed Upper Respiratory

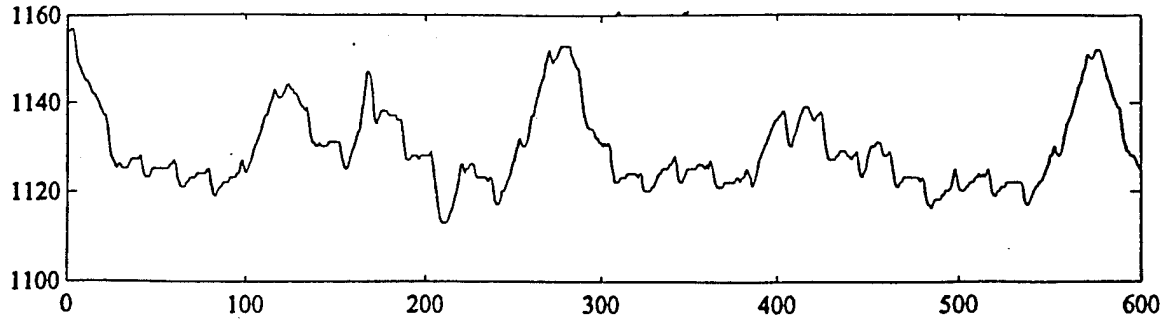


Figure 11 Lower Respiratory

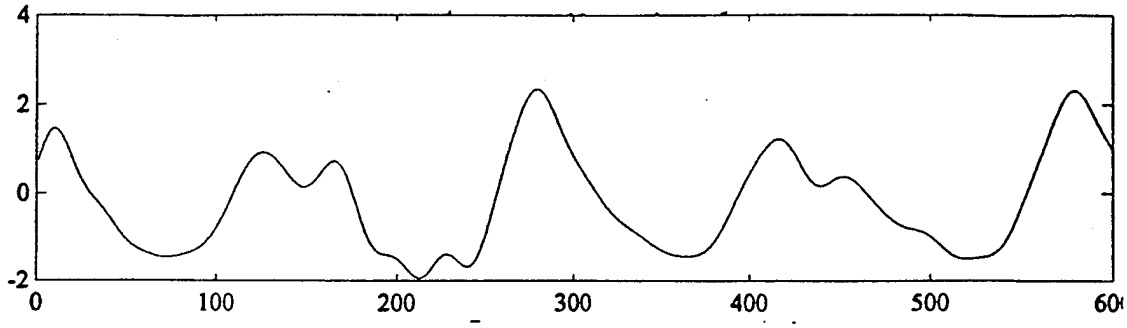


Figure 12 Preprocessed Lower Respiratory

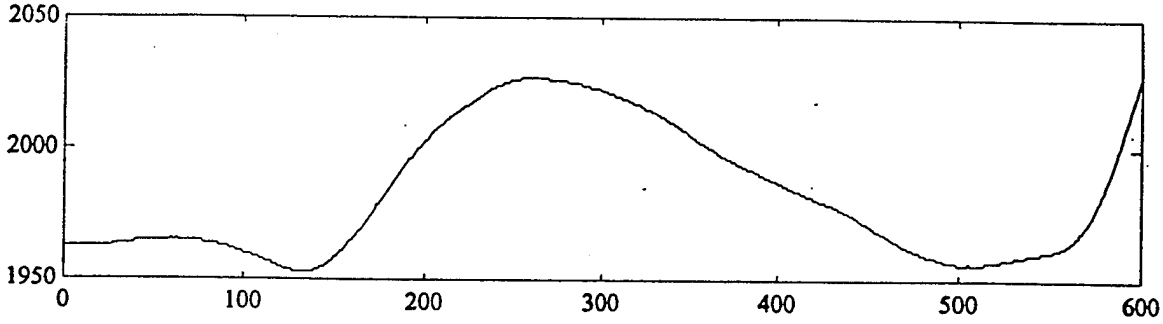


Figure 13 GSR

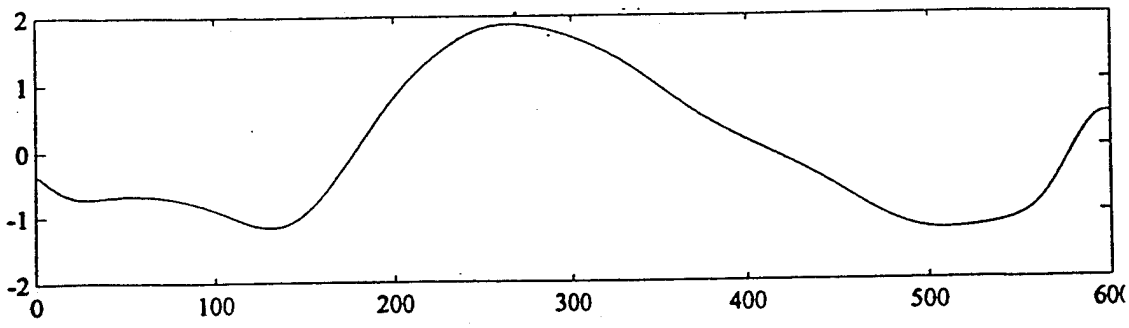


Figure 14 Preprocessed GSR

5.2 Time Domain Feature Extraction

Many of the time domain features were chosen by talking to examiners and finding out what was important to them in an examination [10][11]. One feature examiners use to determine deception involves the height of the peaks in the respiratory signal. If the peaks become smaller or staircase during a relevant question there is a good chance that the subject is being deceptive. From looking at different polygraph charts it could be seen that individual reactions may vary slightly with time. For this reason, many features were extracted from the respiratory channels in order to determine if the deceptive characteristics described above may be present. One feature extracted from the respiratory signal was the average height of the peaks. Because the time fragments from which the features are extracted remain constant, this feature may not give good results for subjects reacting early or late. For this reason, the minimum peak height was also used as a feature.

To try and capture the effect of staircasing, the average of the derivative of the amplitudes of the peaks was used as feature. To compensate for early and late reactions, the maximum of the derivative of the amplitudes of the peaks was also used as a feature.

Another respiratory feature used in this project was the curve length. This feature was successfully used and researched by Howard Timm in the early 1980's[10][13]. Interest in curve length lead to curiosity about the area under the respiratory curve. For this reason it was also extracted to see if it could be used as a feature. Because people tend to breath quicker when they are stressed or nervous, the number of peaks produced during a given period of time was used as a feature.

Because it was one of the first features used to successfully determine deception, Benussi's I/E ratio was tested [3][4]. Benussi's method requires that the I/E ratio of the subject is calculated before and after the examiner asks a question. The value of the I/E ratio calulated after the question is asked is then divided by the value of the I/E ratio before the question is asked. According to Benussi's findings, if the ratio is greater than one, the subject is deceptive. In an attempt to reduce the number of computations required for Benussi's method, a modification of Benussi's feature was tested. In the modification of Benussi's test, the ratio was taken only after the question was asked and was not compared to the subjects I/E ratio before the question was asked.

The examiners we spoke to would usually try to find evidence of deception in respiratory signals first. If a subject did not show a strong respiratory response however, the examiner would analyze the subjects cardiovascular response. Because a subjects heart rate will often increase when deceptive, the number of peaks in the high frequency cardiovascular signal was used as a feature. From looking at many charts, it became evident that some of the processing used in extracting features from the respiratory channels would also be useful in determining deception from the high frequency cardiovascular channel. For this reason, the average of the peak height, minimum of the peak height and curve length were extracted from the high frequency cardiovascular channel in order to determine if they would be useful features.

Many of the standard statistcal features used in other computerized polygraph algorithms were also examined [9]. These features included the mean, the standard deviation, the maximum amplitude, and the minimum amplitude of the signal. Variations

of these such as the minimum subtracted from the maximum were also examined. Although the original use of the curve length and area was to determine deception from the respiratory channel, it was extracted from the GSR and cardiovascular channels as well. It was not possible from looking at the signals to determine if the curve length had changed, but almost any change in a signal would affect this feature. A list of the features extracted from each channel are given in table 5. The programs used to extract these features were written in MATLAB and are included in the appendix of this report.

High frequency cardiovascular

1) mean of signal
2) standard deviation of signal
3) minimum value of signal
4) maximum value of signal
5) curve length of signal
6) area under signal
7) average amplitude of peaks
8) minimum amplitude of peaks
9) derivative of the amplitudes of the peaks in the signal
10) number of peaks in the signal
11) minimum subtracted from maximum

Low frequency cardiovascular

1) mean of signal
2) standard deviation of signal
3) minimum value of signal
4) maximum value of signal
5) curve length of signal
6) area under signal
7) minimum subtracted from maximum

Upper and lower respiratory

1) mean of signal
2) standard deviation of signal
3) minimum value of signal
4) maximum value of signal
5) curve length of signal
6) area under signal
7) average amplitude of peaks
8) minimum amplitude of peaks
9) derivative of the amplitudes of the peaks in the signal
10) number of peaks in the signal
11) inhalation/exhilation ratio
12) ratio of inhalation ratios before and after a question is asked
13) minimum subtracted from maximum

GSR

1) mean of signal
2) standard deviation of signal
3) minimum value of signal
4) maximum value of signal
5) curve length of signal
6) area under signal
7) minimum subtracted from maximum

Derivative of low frequency

1) mean of signal
2) standard deviation ofsignal
3) minimum value of signal
4) maximum alue of signal
5) curve length of signal
6) area under signal
7) minimum subtracted from maximum

Table 5 List of time domain features

5.3 Feature Extraction Methods

To extract the following features which are listed in table 5, (respiratory 7, 8,9,10 ,11 and high frequeny cardiovascular 7, 8, 9), it was necessary to locate the peaks of the respiratory and the high frequency cardiovascular signals. This was not a trivial task because these signals contained low amplitude high frequency noise which was difficult to eliminate without distorting the data (see figures 8,10, and 12). In order to find the useful peaks, two programs were written. The program that found the peaks of the respiratory signal was titled peaklr and the program that found the peaks in the cardiovascular signal titled peakcard. Both programs can be found in the appendix. The way that these programs find peaks is as follows: The second derivative was taken and points that had values equal to zero were labeled as peaks. The amplitudes of the signal at points near these peaks were evaluated and the maximum of these values were labeled as peaks.

In order to eliminate the effects of the low amplitude high frequency noise, it was necessary to check the amplitude of data points that were near each point that had been labeled as a peak. The number of the data points from the peaks that were determined by the second derivative was chosen by examining many respiratory and cardiovascular signals and determining the average width of the peaks in these signals. It was found that twenty points on each side of the each peak found by the second derivative was a satisfactory range for the respiratory signals. Similarly eight points on each side of the initial peak gave would satisfy this criterion for the cardiovascular signal. All of the routines used to perform these operations are in appendix B (see peak.m, peakcard.m, and peaklr.m).

In order to determine the I/E ratio, it was necessary to find the valleys of the respiratory signals as well as the peaks. The method used to find the valleys was the same as that used to find the peaks (see appendix B valley.m and valleylr.m). The I/E ratio was found by the following method. First the time that a valley occurred was subtracted from the time that a peak occurred. Then the time that the peak occurred was subtracted from the time that the next valley occurred. The first value was then divided by the second value (see appendix B ie.m and ieie.m).

6.1 Conclusion

A vector of features was created by the program featurev.m which first executed all of the preprocessing routines. The program then extracted features for all of the questions using the times specified in table 4. This program extracted features from all polygraph files in a directory and produced a set of vectors. These vectors were then used for training and testing of a fuzzy K nearest neighbor classifier. For details on the methods used for training and testing as well as the frequency and correlation domain features used in the study refer to Dastmalchi [14]. For details on the K nearest neighbor algorithm refer to Layeghi [15].

REFERENCES

[1] Dale E. Olsen, et. al., "Recent developments in polygraph testing: A research review and evaluation - A technical memorandum, " Washington,DC: US Government Printing Office 1983.

[2] John C. Kircher and David C. Raskin, "Human versus computerized evaluations of polygraph data in a laboratory setting, " Journal of Applied Psycology, Vol.73, 1988 No2, pp291-308

[3] John E. Reid and Fred E. Inbau, Truth and Deception: The Polygraph (" Lie Dector ") Technique, The Williams & Wilkins Company, Baltimore, Md., 1966

[4] Michael H. Capps and Norman Ansley, "Numerical Scoring of Polygraph Charts: What Examiners Really Do", Polygraph, 1992, 21, 264-320

[5] L. A. Zadeh, "Fuzzy sets", Information and Control, vol. 8, pp.338-332, 1965

[6] James C. Bezdek and Sankar K. Pal, Fuzzy Models for Pattern Recognition Methods that Search for Structures in Data, IEEE Press, Piscataway, NJ. 1992

[7] L. A. Zadeh, "Calculus of fuzzy restrictions," in: L. A. Zadeh,K. S. Fu, K. Tanaka and M. Shimura, eds., Fuzzy Sets and Their Applications to Cognitive and Decision Processes, Academic Press, New York, 1975, pp 1-39

[8] Bart Kosko, Neural Networks and Fuzzy Systems, New Jersey : Prentice-Hall, Inc., 1992.

[9] Brian M. Duston, " Statistical Techniques for Classifying Polygraph Data ", Draft, November 24, 1992

[10] Howard W. Timm, " Analyzing Deception From Respiration Patterns " , Journal of Police Science and Administration, 1982, 1, 47 - 51.

[11] Personal communication with Richard Petty (polygraph examiner), June 1993

[12] Personal communication with Christopher B. Pounds (University of Washington), May 1993

[13] Personal communication with Howard Timm May 1993

[14] Mitra Dastmalchi , " Frequency Domain Features for Pattern Recognition of Polygraph Data", Masters Project, San Jose State University, November 15, 1993

[15] Shahab Layeghi, " Pattern Recognition of Polygraph Data", Masters Project , San Jose State University, November 15, 1993

Appendix A

Preprocessing Programs

```
function y = dercd(var)

% This extracts the derivative of a lowpass
% filtered version of the cardio signal.
%
% To use this command the user must enter the file name
%
% eg.   dercd(variable name)

q = detlc(var);  % detrends the lower frequencies
                 % of the cardio signal

e = diff(q);     % differentiates the lower
                 % frequencies of the cardio signal

x = e/std(e);

y = [x',x(length(x))']';
```

```
function y = detgsr(var)

% This function detrends the gsr
%
% To use this command the user must enter the file name
%
% eg.   detgsr(file name)

dtrnd = detrend(var(:,1));
                          % elliminates dc trends in signal
                          % eg. a line added to the signal
window = 100;

dtrnd = [dtrnd', zeros(window/2 - 1,1)']';
                  % adds zeros to end of signal so that no
                  % information is lost during filter delay


b = fir1(window,.03);
x = filter(b,1,dtrnd);
q = x/std(x);
l = length(q);

y = q(window/2:l);              % compensate for time delay
```

```
function y = dethic(var)

% This function detrendeds the high frequencies
% of the cardio signal.
%
% To use this command the user must enter the file name
%
% eg.   dethic(file name)

dtrnd = detrend(var(:,2));  % elliminates dc trends in signal
                            % eg. a line added to the signal

window = 134;

dtrnd = [dtrnd', zeros(window/2 - 1,1)']';
                        % adds zeros to end of signal so that no
                        % information is lost during filter delay


b = fir1(window,.035,'high');
                        % filter to elliminate low frequencies
x = filter(b,1,dtrnd);
q = x/std(x);

l = length(q);

y = q(window/2:l);          % compensate for time delay
```

```
function y = detlc(var)

% This function extracts and detrends the low
% frequencies of the cardio signal
%
% To use this command the user must enter the file name
%
% eg.   detlc(file name)

dtrnd = detrend(var(:,2));  % elliminates dc trends in signal
                            % eg. a line added to the signal
window = 134;

dtrnd = [dtrnd', zeros(window/2 - 1,1)']';
                        % adds zeros to end of signal so that no
                        % information is lost during filter delay

b = fir1(window,.035);    % filter to elliminate high frequencies
x = filter(b,1,dtrnd);
q = x/std(x);

l = length(q);

y = q(window/2:l);        % compensate for time delay
```

```
function y = detlr(var)

% This function extracts and detrends the lower respiratory signal
%
% To use this command the user must enter the file name
%
% eg.   detltr(file name)

dtrnd = detrend(var(:,4)); % elliminates dc trends in signal
                           % eg. a line added to the signal
window = 240;

dtrnd = [dtrnd', zeros(window/2 - 1,1)']';
                          % adds zeros to end of signal so that no
                          % information is lost during filter delay


b = fir1(window,.083);    % filter to elliminate noise
x = filter(b,1,dtrnd);
q = x/std(x);

l = length(q);

y = q(window/2:l);         % compensate for time delay
```

```
function y = detur(var)

% This function detrends the upper respiratory signal
%
% To use this command the user must enter the file
%
% eg.   detur(file name)

dtrnd = detrend(var(:,3));  % elliminates dc trends in signal
                            % eg. a line added to the signal

window = 240;

dtrnd = [dtrnd', zeros(window/2 - 1,1)']';
                          % adds zeros to end of signal so that no
                          % information is lost during filter delay


b = fir1(window,.08);              % filter to elliminate noise
x = filter(b,1,dtrnd);
q = x/std(x);

l = length(q);

y = q(window/2:l);                  % compensate for time delay
```

Appendix B

Feature Extraction Programs

```
function [x,y,z] = featurev(file_name,relevant,irrelevant,control,features)

% This function produces a feature vector for a given file
% Relevent, irrelevent, and control are vectors which contain
% the questions these features are extracted from.
%
% eg. featurev(t79,[3 5],[1 4], [6 10],feature_list)

% The above example gives the features for
% the file t79 of the 3rd and 5th question which are relevent in this
% MGQT format, the 1st and 4th question which are irrelevent
% and the 6th and 10th questions which are control

% feature_list=['10mean(frag ) ';
%               '20curve(frag )';
%               '30area(frag ) '];


feature_list = features



% The channels are ordered as follows:
% 1:GSR, 2:HiCardio, 3:LowCardio, 4:DerLowCardio, 5:LowResp, 6:UpResp


% This is a matrix of the time delay after asking a question to start of extracting
% the feature, and finish extracting the feature for each channel.

Times=[ 2, 14;
        3, 9 ;
        2, 18;
        0, 8 ;
        2, 18;
        2, 18];

% These are preprocessing functions.
Preprocess=[    'detgsr';
                'dethic';
                'detlc ';
                'dercd ';
                'detlr ';
                'detur '];

data=zeros(6,length(file_name(:,5)));
% Standardize and detrend the channels and derive new channels

for i=1:6,
        data(i,:)=eval([Preprocess(i,:),'(file_name)'])';
end
```

```
marker = file_name(:,5);    % 0 begin test and end test
                            % 0 examiner begins asking question
                            % 1 examiner finishes asking question
                            % 2 subject begins response to question
                            % 9 does not mark an event

begin = find(marker == 0);     % finds indecies where marker = 0 (question begins)
begin=begin(2:length(begin));      % elliminates the marker at the beginning of the test

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This for loop creates feature vectors for each relevant quesion
%
% eg x = [mean(gsr),std(gsr),area(gsr),mean(lr),std(lr),area(lr),etc.....
%          curve length,amplitude of peaks,# of peaks]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

feature_count=1;

for i = 1:length(relevant),
        question=relevant(i);

        for j=1:length(feature_list(:,1))
                channel_number=eval(feature_list(j,1));
                second_channel=eval(feature_list(j,2));
                st=begin(question)+30*Times(channel_number,1);
                fn=begin(question)+30*Times(channel_number,2);
          st2=begin(question)-30*Times(channel_number,2);
          fn2=begin(question)-30*Times(channel_number,1);
                fr=feature_list(j,3:length(feature_list(1,:)));
                frag=data(channel_number,st:fn);
                frag2 = data(channel_number,st2:fn2);
                if second_channel ~= 0
                        st3=begin(question)+30*Times(second_channel,1);
                fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                end
                tempy=eval(fr);
          for m = 1:length(tempy)
             x(feature_count) = tempy(m);
                   feature_count=feature_count+1;
          end
        end
end
%-------------------------------------------------------------------
% Irrelevant questions

feature_count=1;

for i = 1:length(irrelevant),
        question=irrelevant(i);

        for j=1:length(feature_list(:,1))
                channel_number=eval(feature_list(j,1));
```

```
                second_channel=eval(feature_list(j,2));
                st=begin(question)+30*Times(channel_number,1);
                fn=begin(question)+30*Times(channel_number,2);
          st2=begin(question)-30*Times(channel_number,2);
          fn2=begin(question)-30*Times(channel_number,1);
                fr=feature_list(j,3:length(feature_list(1,:)));
                frag=data(channel_number,st:fn);
                frag2 = data(channel_number,st2:fn2);
                if second_channel ~= 0
                        st3=begin(question)+30*Times(second_channel,1);
                fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                end
                tempy=eval(fr);
          for m = 1:length(tempy)
             y(feature_count) = tempy(m);
                   feature_count=feature_count+1;
          end
        end
end

%-----------------------------------------------------------------
% Control questions

feature_count=1;

for i = 1:length(control),
        question=control(i);

        for j=1:length(feature_list(:,1))
                channel_number=eval(feature_list(j,1));
                second_channel=eval(feature_list(j,2));
                st=begin(question)+30*Times(channel_number,1);
                fn=begin(question)+30*Times(channel_number,2);
          st2=begin(question)-30*Times(channel_number,2);
          fn2=begin(question)-30*Times(channel_number,1);
                fr=feature_list(j,3:length(feature_list(1,:)));
                frag=data(channel_number,st:fn);
                frag2 = data(channel_number,st2:fn2);
                if second_channel ~= 0
                        st3=begin(question)+30*Times(second_channel,1);
                fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                end
                tempy=eval(fr);
          for m = 1:length(tempy)
             z(feature_count) = tempy(m);
                   feature_count=feature_count+1;
          end
        end
end
```

```
function y = ampcard(var)

% This function finds the average of the amplitudes
% of the peaks in the high
% cardio signal over a specified period of time.
%
% To use this command the user must enter the
% file name and the start and finish points
% of the signal to be displayed
%
% eg.   ampcard(variable name)

p = peakcard(var);       % the indecies of the peaks

for n = 1:length(p)

   q(n) = var(p(n));     % amplitude of the peaks

end

y = sum(q)/length(q);
```

```
function y = ampr(var)

% This function finds the average of the
% amplitudes of the peaks in the lower
% respiratory signal over a specified period of time.
%
% To use this command the user must
% enter the variable name
%
% eg.   ampr(variable name)

p = peaklr(var);        % the indecies of the peaks

for n = 1:length(p)

   q(n) = var(p(n));    % amplitude of the peaks

end

y = sum(q)/length(q);
```

```
function y = curve(var)

% This function finds the length of the variable
%
% To use this command the user must enter the
% variable name and the start and finish points
% of the signal to be displayed
%
% eg.   curve(variable name)


x = sqrt(diff(var).^2 + 1);
y = sum(x);
```

```
function y = ie(var)

% This function takes the i/e ratio of the respiratory signals.
%
% To use this command the user must enter the variable name
%
% eg.   ie(variable name)

p = peaklr(var);        % finds the indices of
                        % the peaks in a signal and puts them
                        % in a vector a
plength = length(p);

v = valleylr(var);       % finds the indices of the
                         % valleys in a signal and puts them
                         % in a vector b
vlength = length(v);

if vlength < 2 | plength < 2        % check that enough peaks
                                    % and valleys exist for
                                    % the calculation to be done

   message = ' Warning !!!! Not enough data'

end

if p(1) > v(1)

   for n = 1:vlength - 1

       q = p(n) - v(n);             % calculates a vector of
                                    % e/i ratios for the given
                                    % time period

       z = v(n + 1) - p(n);

       e(n) = q ./ z;
   end

end

if p(1) < v(1)

   for n = 1:vlength - 1

       q = p(n + 1) - v(n);     % calculates a vector of
                                % e/i ratios for the peaks
                                % and valleys in the
                                % given time period

       z = v(n + 1) - p(n + 1);
```

```
        e(n) = q ./ z;
    end

end

y = mean(e);
```

```
function y = ieie(var1,var2)

% This function takes the i/e ratio of the respiratory signals
% before and after a question is asked.  It then divides the two
% values.
%
% To use this command the user must enter the variable name
%
% eg.   ieie(variable name1, variable name2)

a = ie(var1);

b = ie(var2);

y = a/b;
```

```
function y = peak(var)

% This function finds the peaks in a signal and returns the index
% It also creates a plot of the variable with the peaks marked
%
% To use this command the user must enter the variable name
% of the signal to be displayed
%
% eg.   peak(variable name)

q = diff(var);      % differentiates the variable

z = q>0;            % z = 1 if q is greater than 0

f = diff(z);        % 2nd derivative of the variable

a = f<0;

y = find(a);        % finds the indices where the 2nd derivative
                    % is -1 which indicates peak
```

```
function y = peakcard(var)

% This function finds the peaks in
% the cardio signal and returns a vector of
% indexes where they occur.
%
% To use this command the user must enter the variable name
%
% eg.   peakcard(variable name)


ty = peak(var);

if ty(1) < 8
        ty = ty(2:length(ty));
end

if ty(length(ty)) > length(var) - 8
        ty = ty(1:length(ty)-1);
end

for n = 1:length(ty);
                          % finds the maximum peak over a 10 point s
pan


     temp = var(ty(n)-8 : ty(n)+8);

     z(n) = ty(n) - 9 + find(temp == max(temp));
                                % finds the time that the peak
                                % occurs in the original signal
end

for n = 1:length(z)-1  % elliminates duplicate indicies

    if z(n) == z(n+1)

          z(n) = 0;

     end

end

ind = find(z);              % finds indecies of elements
                            % that are not equal to zero

for n = 1:length(ind)       % elliminates 0 elements

     z(n) = z(ind(n));

end
```

```
y = z(1:length(ind));

% pmark = zeros(1,length(var)); % a vector of 1's where peaks occu
r
                                % 0's everywhere else
% pmark(y) = ones(1,length(y));

% plot(var,'r')

% title('lr marked with peaks')

% hold on

% plot(5*pmark,'g')

% hold off
```

```
function y = peaklr(var)

% This function finds the peaks
% in the lr signal and returns a vector
% of indecies where they occur.
%
% To use this command the user must enter the variable name
%
% eg.   peaklr(variable name)


[b,a] = butter(4,.034);                % elliminate noise
filtout = filtfilt(b,a,var);

ty = peak(filtout);  % finds the time that the
                     % peaks of filtered lr signal occur

if ty(1) < 20
        ty = ty(2:length(ty));
end

if ty(length(ty)) > length(var) - 20
        ty = ty(1:length(ty)-1);
end

for n = 1:length(ty)

     temp = var(ty(n)-20:ty(n)+20);
     z(n) = ty(n) - 21 + find(temp == max(temp));
                         % finds the time that the peak occurs in
                         % the original signal
end

for n = 1:length(z)-1              % elliminates duplicate indicies

    if z(n) == z(n+1)

          z(n) = 0;

     end

end

ind = find(z);             % finds indecies of elements
                           % that are not equal to zero

for n = 1:length(ind)  % elliminates 0 elements

     z(n) = z(ind(n));

end
```

```
y = z(1:length(ind));
```

```
function y = peaknumc(var)

% This function finds the number of
% peaks in the high cardio signal
%
% To use this command the user
% must enter the variable name
%
% eg.   peaknumc(variable name)

p = peakcard(var);          % the indecies of the peaks

y = length(p);
```

```
function y = peaknumr(var)

% This function finds the number
% of peaks in the respiratory signal
%
% To use this command the user
% must enter the variable name
%
% eg.   peaknumr(variable name)

p = peaklr(var);          % the indecies of the peaks

y = length(p);
```

```
feature_list=[   '10mean(frag)           ';
                 '10curve(frag)          ';
                 '10area(frag)           ';
                 '20mean(frag)           ';
                 '20curve(frag)          ';
                 '20area(frag)           ';
                 '20ampcard(frag)        ';
                 '20peaknumc(frag)       ';
                 '30mean(frag)           ';
                 '30curve(frag)          ';
                 '30area(frag)           ';
                 '40mean(frag)           ';
                 '40curve(frag)          ';
                 '40area(frag)           ';
                 '50mean(frag)           ';
                 '50curve(frag)          ';
                 '50area(frag)           ';
                 '50ampr(frag)           ';
                 '50peaknumr(frag)       ';
                 '50ie(frag)             ';
                 '50ieie(frag, frag2)    ';
                 '60mean(frag)           ';
                 '60curve(frag)          ';
                 '60area(frag)           ';
                 '60ampr(frag)           ';
                 '60peaknumr(frag)       ';
                 '60ie(frag)             ';
                 '60ieie(frag, frag2)    '];
[x y z] = featurev(t79,[1 2],[3 4],[6 10],feature_list)
```

```
function y = valcard(var,start,finish)

% This function finds the valleys in
% the lr signal and returns a vector of indexes where
% they occur
%
% To use this command the user must enter the
%  file name and the start and finish points
% of the signal to be displayed
%
% eg.   valcard(file name, start, finish)


k = hicardio(var,start,finish);


[b,a] = butter(4,.034);        % elliminate high frequencies
filtout = k; % filtfilt(b,a,k);

ty = valley(filtout,start,finish) % finds the time that the
                                  % peaks of filtered lr signal oc
cur

l = length(ty);


for n = 1:l

     temp = k(max(1,ty(n)-10+start) : min(ty(n)+10+start,length(k)
));

     if ty(n)<10
        dd=length(temp)/2+1;
     else
        dd=11;
     end

     y(n) = ty(n) - dd + find(temp == min(temp));
                         % finds the time that the peak occurs in
                         % the original signal
end

vmark = zeros(1,finish - start); % a vector of 1's where peaks occ
ur
                                 % 0's everywhere else
vmark(y) = ones(1,length(y));

subplot(211),plot(k(start:finish),'r')
```

```
title('lr marked with peaks')
hold on
plot(-5*vmark,'g')
hold off
subplot(212),plot(filtout(start:finish),'r')
title('filtered lr marked with peaks')
hold on
plot(vmark,'g')
hold off
% subplot(223),plot(k(start:finish),'r')
% hold on
% plot(5*a(1:finish - start - 3),'g')
% hold off
% subplot(224),plot(x)
% subplot(111)
```

```
function y = valley(var)

% This function finds the
% valleys in a signal and returns the index

% To use this command the user
% must enter the variable name
%
% eg.   valley(variable name)

q = diff(var);          % differentiates the variable

z = q > 0;      % z = 1 if q is greater than 0

f = diff(z);    % 2nd derivative of variable

a = f > 0;      % finds valleys

y = find(a);    % finds the indices where the 2nd derivative
                % is +1 which indicates valleys
```

```
function y = valleylr(var)

% This function finds the valleys in
% the lr signal and returns a vector of
% indecies where they occur
%
% To use this command the user must enter the variable name
%
% eg.   valleylr(variable name)


[b,a] = butter(4,.034);        % elliminate high frequencies
filtout = filtfilt(b,a,var);

ty = valley(filtout);    % finds the time that the
                         % peaks of filtered lr signal occur

for n = 1:length(ty)

     temp = var(max(1,ty(n)-20) : min(ty(n)+20,length(var)));

     if ty(n)<20
        dd=length(temp)/2+1;
     else
        dd=21;
     end

     z(n) = ty(n) - dd + find(temp == min(temp));
                         % finds the time that the peak occurs in
                         % the original signal
end

for n = 1:length(z)-1    % elliminates duplicate indicies

    if z(n) == z(n+1)

          z(n) = 0;

     end

end

ind = find(z);           % finds indecies of elements
                         % that are not equal to zero

for n = 1:length(ind)    % elliminates 0 elements

     z(n) = z(ind(n));

end
```

```
y = z(1:length(ind));
```

Report No. DoDPI96-R-0002

Features Analysis of the Polygraph

Mitra Dastmalchi
San Jose State University
Department of Electrical Engineering
San Jose, CA 95106

December 1993

Department of Defense Polygraph Institute
Fort McClellan, AL 36205

Table of Contents

List of Charts

List of Figures

List of Tables

Acknowledgement

I express my sincere appreciation to all of those who have contributed to this project. Special recognition goes to my advisor, Dr. Ben Knapp, for his advise and encouragement. I am also grateful for the help of my partner, Shahab Layeghi and Eric Jacobs. Expecially Shahab, for his support and valuable suggestions.

0 Introduction

The polygraph examination is one of the most popular methods to measure deception. Polygraph tests are used in criminal investigations to determine if a suspect is being deceptive when answering the questions concerning a crime. During a polygraph test, the subject is asked a series of control, relevant and irrelevant questions that provide physiological responses for comparison with question that are relevant to the investigation. The three physiological responses that are currently measured are electrocardiogram, galvanic skin response and respiration. The controversy surrounding the use of polygraph tests centers on the subjective judgment of polygraph examiners in classifying the subject as deceptive or non-deceptive. The object of this project is to develop an automatic scoring system to overcome this perception. The computer algorithm will be able to use more sophisticated techniques than human examiners, should be more accurate and will ensure consistency from case to case.

In order to implement the automatic scoring system, two main algorithms were developed. These were: the feature extraction algorithm, which process the polygraph data in three time, correlation and frequency domains, and the fuzzy classifier algorithm, which accepts the features and determines the possibility of deception. Because of the nature of the input, fuzzy logic was chosen to implement the system which gives the possibility of belonging of an input to each class. Initially, a set of features based on physiological reactions were selected. Then, the fuzzy K-nearest neighbor classifier was used to classify the features.

1 Polygraph

1.1 Polygraph Examination

The primary use of the polygraph test is during the investigation stage of the criminal justice process. In addition to the significance role in criminal justice, they are also used for national security, intelligence and counterintelligence activities [1]. The three physiological responses currently obtained from a polygraph examination are electrocardiogram, respiration and galvanic skin response. Electrocardiogram is measured by placing a standard cuff on the arm over the brachial artery. Respiration is monitored by placing rubber tubes around the abdominal area of the subject. Skin conductivity is measured by electrodes placed on two fingers of the same hand of the subject [1].

The effectiveness of a polygraph examination is often the result of the test format that is used. A polygraph test format is an ordered combination of relevant question about an issue, control questions that provide physiological responses for comparison and irrelevant questions that act as a buffer [1]. An example or a relevant question is, " did you embezzle any of the missing $12000?" The corresponding control question would be about stealing; an example is, "did you ever steal money or property from an employer?" The example of an irrelevant question is, " is your name John?" Irrelevant questions are answered truthfully and are not stressful. The rational for scoring these tests is that a deceptive subject will be more threatened by the relevant question than by the control question while a non deceptive subject will be more threatened by the control questions than the relevant question.

Polygraph charts are usually analyzed by a human interpreter for evidence of truth or deception. A control question polygraph chart usually consists of 3 sets of control relevant question pairs separated by neutral questions. The examiner scores the charts by comparing each relevant question. For each of three physiological responses, he will give a numerical score ranging from -3 t0 +3, depending on the magnitude of the difference. He then adds up scores for all control relevant pairs. If the score is below threshold value, he scores the chart as deceptive or non deceptive.

Sometimes the examiner can not make a clear decision and must score the chart as inconclusive. The examiner's decision will be based on his or her experience and training. For example, a change in the polygraph tracing considered by one examiner as a physiological changes, may be considered by another as an artifact of the recording system. In an effort to eliminate the inconsistencies involved in interpreting polygraph data, computer algorithm are being developed.

1.2 History[1]

The first attempt to use a scientific instrument in an effort to detect deception occurred around 1895 [2]. That was the year that Cesar Lombroso published the results of his experiments in which a hydrosphygmograph was used to measure the blood pressure-pulse changes of criminals in order to determine whether or not they were deceptive. Although the hydrosphygmograph was originally intended to be used for medical purposes, Lombroso found that it worked well for lie detection. Lombroso may have been the first to use a peak of tension test format. This was done by showing a suspect a series of photographs of children, one being the victim of sexual assault. If the suspect did not react more to the victims picture than the pictures of the other children, Lombroso concluded that the suspect did not know what the victim looked like and therefore was not the alleged perpetrator.

In 1914 Vittorio Benussi published his research on predicting deception by measuring recorded respiration tracings [3]. He found that if the length of inspiration were divide by the length of expiration, the ratio would be larger after lying than before lying and also before telling the truth than after telling the truth. In 1921 John A. Larson constructed an instrument capable of simultaneously recording blood pressure pulse and respiration during an examination [2][3]. Larson reported accurate results which prompted Leonarde Keeler to construct a better version of this instrument in 1926 [2][3].

The use of galvanic skin response in lie detection began during the turn of the century. It's usefulness, however, did not become evident until the 1930's during which time several articles written by Father Walter G. Summers of Fordham University in New York [3]. In these articles he reports over 90 criminal cases in which examination using the galvanic skin response had all been successful and confirmed by confession or supplementary evidence. The usefulness of the galvanic skin response prompted Keeler to add an galvanometer to his polygraph. At the time of Keelers death in 1949, the Keeler Polygraph recorded blood pressure-pulse, respiration, and galvanic skin response [3].

1.3 Modern Test Formats[1]

The effectiveness of a polygraph examination is often the result of the test format that is used. A polygraph test format consists of an ordered combination of relevant questions about an issue, control questions that provide a physical response for comparison, and irrelevant questions that also provide a response or the lack of a response for comparison [1][3]. Three general types of test formats are in use today. These are Control Question Tests, Relevant-Irrelevant Tests, and Concealed Knowledge Tests. Each of the general test formats may have a number of more specific variations. Each test consists of two to

[1]These sections were exerpted from Jacobs [10].

five charts containing a prescribed series of questions. The test format that is used in an examination is determined by the test objective [2][3].

The concealed knowledge test, also called peak of tension test, is used when facts about a crime are known only by the investigators and not by the public. In this case, a subject would not know the facts unless he or she was guilty of the crime. For example, if a gun was used in a crime and the public did not know the caliber, an examiner could ask a suspect if it was a 22 caliber, a 38 caliber, or a 9 mm. If the gun used was a 9 mm and the suspect was deceptive, a polygraph chart would probably indicate evidence of deception.

A control question test is often used in criminal investigations. Relevant-Irrelevant tests are usually used to test people trying to obtain security clearance or get a job. In this test, relevant questions are compared to irrelevant questions. Very few control questions are asked. The purpose of control questions in this test is to make sure that the subject is capable of reacting at all.

1.4 Present Day Equipment[2]

The most popular polygraph machines today are the Reid Polygraph developed in 1945 and the Axciton Systems computerized polygraph developed in 1989 [1][4]. The Reid polygraph scrolls a piece of paper under pens that record the biological signals. The Axciton polygraph digitizes physiological signals and uses a computer to process them. The sampling frequency of the Axciton machine is 30 Hz. Axciton provides a computer based system for ranking the subject responses but allows printouts of the charts to be scored by hand the traditional way.

Both machines record the same biological signals using standard methods. Blood pressure is measured by placing a standard blood pressure cuff on the arm over the brachial artery. Respiration is monitored by placing rubber tubes around the abdominal area and the chest of the subject. This results in two signals, an upper and lower respiratory signal. Skin conductivity is measured by placing electrodes on two fingers of the same hand.

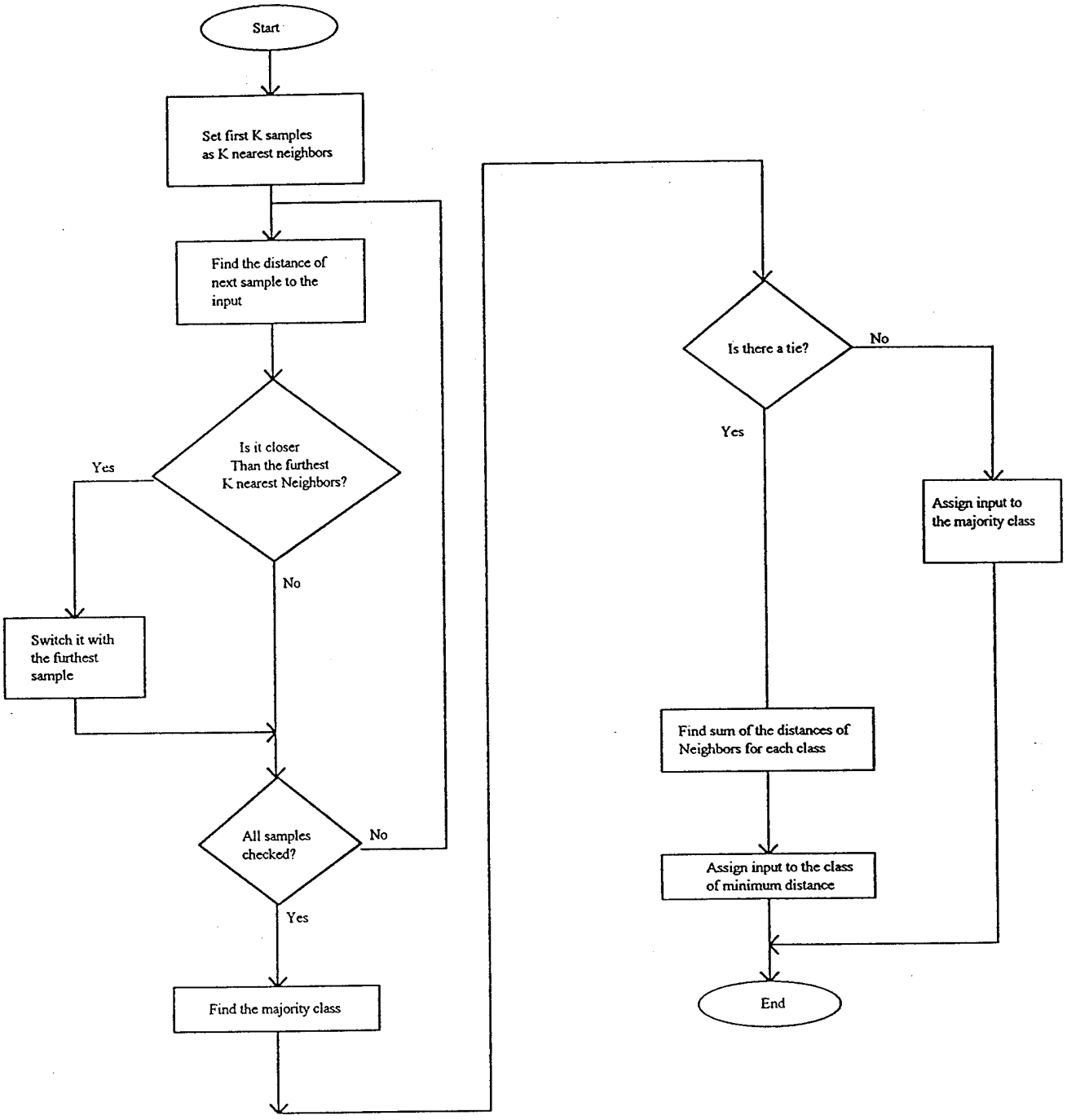
[2]This section was exerpted from Jacobs [10].

2 Classifier Algorithm

2.1 K-Nearest Neighbor Algorithm[3]

K-nearest neighbor algorithm is a supervised classification method. There is no need for the training or adjusting the classifier. A set of labeled input samples is given to the classifier. When a new sample is given to the system, it finds its K nearest neighboring samples, and assigns this sample to the class that the majority of the neighbors belong to. K could be any positive integer. When K is set to 1, the algorithm is called the nearest neighbor algorithm. In this case each new sample is assigned to the class of its nearest neighbor. If K is greater than 1, it is possible that there is no majority class. To remove this tie, the sum of the distances of the new sample to its neighbors in each class is computed and the sample is assigned to the class that has the minimum distance. The main advantage of using this method is that the samples of each class are not needed to cluster in a pre specified shape. For example, for a two class classification, the K-nearest neighbor classifier can still give very good results if the samples of each class are clustered in two distinct points in the space. The algorithm for the K nearest neighbor is shown in flow chart 1. It is supposed that C is the number of classes, K is the number of neighbors in KNN, $x_i x_i$ is the *ith* labeled sample and y is the input to be classified.

[3]This section was exerpted from Layeghi [11].



Flow chart 1. Fuzzy K Nearest Neighbor Algorithm

The fuzzy K nearest neighbor algorithm uses the same idea of conventional K nearest neighbor algorithm, that is finding the K samples that are closest to sample to be classified. But there is a conceptual difference in classification. When fuzzy classification is used, the input is not assigned to a single class. Instead, the degree of belongings of the input to each class is determined by the classifier. By using this method more information is obtained about the input. For example if the result of classification determines membership of an input to class A is 0.9 and to class B is 0.1, it means the input belongs to class A with a very good possibility. But if the membership to class A is 0.55 and to class B is 0.45, it means that we cannot be very sure about the classification of the input. If the crisp classifier is used, in both cases the input will be assigned to class A and no further information is obtained.

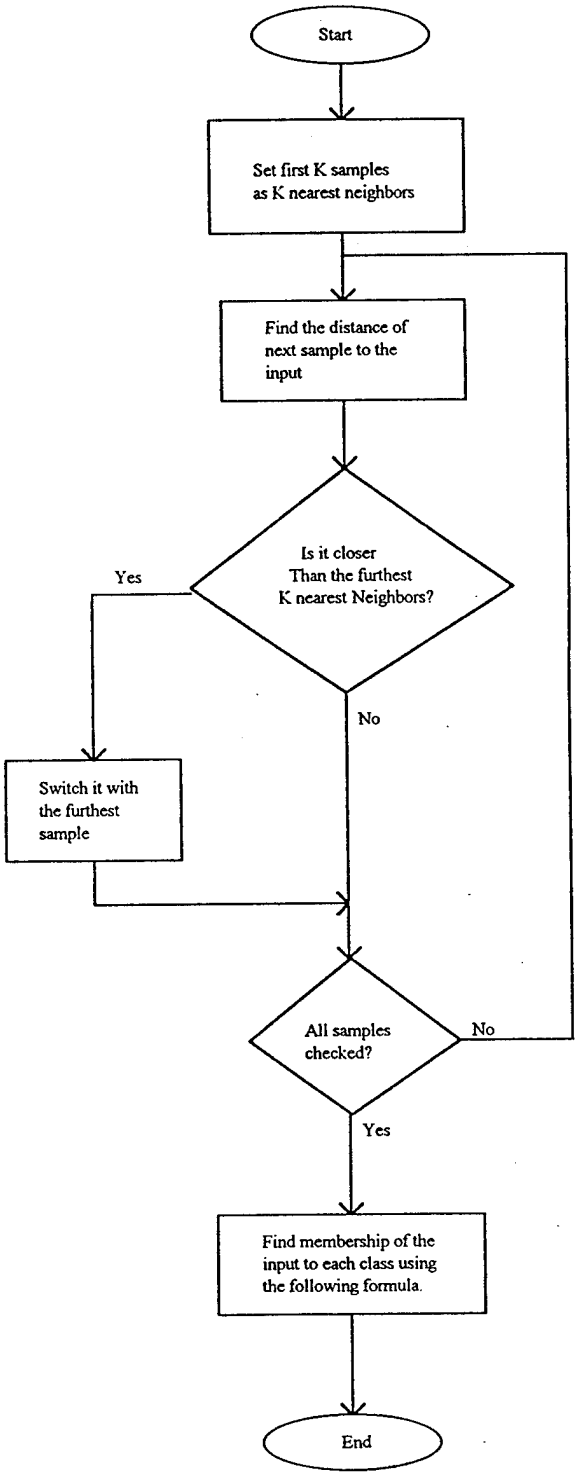
Refer to [5] [6] for more detailed discussions about fuzzy K nearest neighbor algorithms. The flowchart for a fuzzy K nearest neighbor classifier is drawn in flow chart 2.

The first step in the fuzzy K nearest neighbor algorithm is the same as first step in crisp classifier. In both cases K nearest neighbors of the input are found. While in crisp classifier the majority class of the neighbors is assigned to the input, in Fuzzy classifier membership of the input to each class should be found. In order to do so the membership vector of each sample is combined to obtain the membership vector of the input. If the samples are crisply classified, membership vectors should be assigned to them. One method to do so is to assign the membership of 1 to the class that it belongs to, and membership of 0 to other classes. Other methods assign different memberships to the samples according to its distance from the mean of the class, or the distances from the nearby samples of its own class and the other classes.

When the membership vectors of the labeled samples are specified, they are combined to find the membership vector of the unknown class. This procedure should be done in a way that samples that are closer to the input have more effect on the resultant membership function. The following formula uses the inverse distance to weigh the membership functions. x is the input to be classified, x_j is the j*th* nearest neighbor and u_{ij} is the membership of the j*th* nearest neighbor of the input in class i. D(x,y) is a distance measure between the vectors x and y which could be the Euclidean distance.

$$u_i(x) = \frac{\sum_{j=1}^{K} u_{ij}\left(1/D(x,x_j)^{\frac{1}{m-1}}\right)}{\sum_{j=1}^{K}\left(1/D(x,x_j)^{\frac{1}{m-1}}\right)}$$

m is a parameter that changes the weighing effect of the distance. When m >> 1, all the samples will have the same weight. When m approaches 1, nearest samples have much more effect on the membership value of the input.



Flow chart 2. Fuzzy k nearest neighbor

$$u_i(x) = \frac{\sum_{j=1}^{K} u_{ij}\left(1/D(x,x_j)^{\frac{1}{m-1}}\right)}{\sum_{j=1}^{K}\left(1/D(x,x_j)^{\frac{1}{m-1}}\right)}$$

3 Frequency and correlation Domain Features

3.1 Preview

The purpose of this chapter is to show how the frequency and correlation domain representations of polygraph signals can be used effectively in polygraph analysis. The first step in analysis of a time series is to plot the data and to obtain simple descriptive measures of the main properties of the series. For some series, in addition to features such as trend, seasonal effect and cyclic changes, more sophisticated features such as mean, variance, auto correlation and frequency content will be required to provide an adequate analysis.

Most physical processes, including polygraph signals, involve a random element in their structures. Currently, human examiners score polygraph tests by analyzing obvious features in the time domain. It is presumed that processing polygraph signals in frequency and correlation domain will provide features which are discriminator between deceptive and non-deceptive subjects. Before finding the frequency domain features the trend in the electrocardiogram channel was eliminated. In order to do so, a high frequency electrocardiogram channel, called heart pulse, is produced by highpass filtering it.

The goal of this chapter is to explain the techniques used to extract appropriate features in frequency and correlation domains. The methods for estimating features of the polygraph signals such as fundamental frequency, spectral density and cross correlation between the channels will be discussed.

3.2 Fundamental Frequency

One feature which is considered important in the frequency domain is the fundamental frequency of the signal. The purpose of finding the fundamental frequency is to classify the way the frequency changes in a specific time segment. The assumption in polygraph signals is that the frequency of the signal changes after a relevant or a control question is asked. Different methods have been proposed to find the fundamental frequency of a signal. One of these methods is using the auto correlation function.

The auto correlation representation of a signal is a convenient way of displaying certain properties of the signal. For example, the auto correlation function of a periodic signal is also periodic with the same period. For periodic signals with period P, the auto correlation function attains a maximum at samples $0, \pm P, \pm 2P, \ldots$. Regardless of the time origin of the signal, the period can be estimated by finding the location of the first maximum in the auto correlation function [7].

This property makes the auto correlation function an attractive basis for estimating periodicity in most signals including the electrocardiogram and respiration signals of the polygraph records. Therefore, a short segment of the signals (electrocardiogram and respiratory) after each question is selected and pre-processed. The auto correlation is then calculated for the windowed segments of the heart pulse and respiratory signals using MATLAB. Figure 1 shows the examples of auto correlation functions computed for heart pulse with N = 150 and upper respiratory with N = 400 sampled at 30 Hz. N is the number of samples .

It is noticeable that the auto correlation functions of the above signals are a mixture of damped exponential and sinusoids. For the heart pulse, peaks occur approximately at multiples of 20 samples indicating a period of 20/30=0.67 seconds or a fundamental frequency of approximately 1.5 Hz. For the upper respiratory, peaks occur approximately at multiples of 133 samples indicating a period of 133/30 = 4.4 seconds or a fundamental frequency of approximately 0.23 Hz.

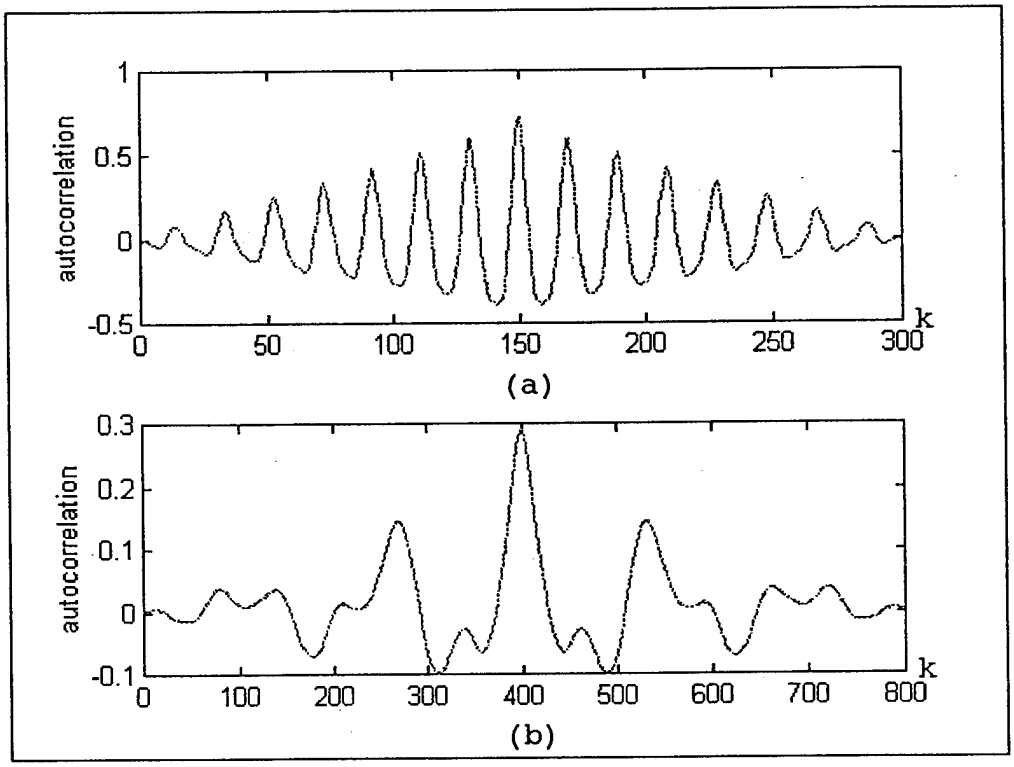


Figure 1. Plots of auto correlation function for (a) heart pulse and (b) upper respiratory where k is the number of samples.

For some subjects, the period of the electrocardiogram or upper respiratory signal changes across the N sample interval. Also, the shape of the signal varies somewhat from period to period. Because of the finite length of segments involved in the computation of auto-correlation, there is less and less data involved in the computation as the lag increases. This leads to the reduction in amplitude of the correlation peaks as lag increases.

An important issue is how N should be chosen to give a good indication of periodicity. Because we are interested in observing changes in signal after the question is asked, N should be small. On the other hand, it should be noted that to get any indication of periodicity in the auto correlation function, the window must have the duration of at least two periods of the waveform. In order to choose the best N, the fundamental frequency for different time frames without overlap were calculated and the results were examined. The fundamental frequencies of heart pulse for the four second frame are shown in Table 1 and 2 in Appendix A. No single value of N is entirely satisfactory because the frequency changes from individual to individual. However, a suitable practical choice for N was chosen on the order of 180 and 480 for heart pulse and upper respiratory respectively.

3.3 Modeling

Detailed information about a time series can be obtained from creating a model. In this section a model will be found for the heart pulse signal. Finding a suitable model for a given time series depends on the properties of the series and the number of observations available. In signal modeling the output signal is known and the model development is based upon the fact that signal points are correlated. Estimated auto correlation function (ACF) of the time series is helpful in identifying which type of ARMA model is appropriate and gives the best representation of the signal.

The ACF of a MA process cuts off at lag q whereas the ACF of an AR process is a mixture of damped exponential and sinusoids and dies out slowly. For example, if r1 is significantly different from zero but the subsequent values of r_k are all close to zero then an MA(1) model is indicated since its theoretical ACF is of this form. Alternatively, if r_1, r_2, r_3 ,.. appear to be decreasing exponentially, then an AR(1) model may be appropriate.

It is usually difficult to find the order of an AR process from the sample ACF alone. A model with too low an order will not represent the properties of the signal. Also a model with too high an order will represent any measurement noise or inaccuracies. Therefore, neither a high order nor a low order model will be a reliable representation of the signal. As a result, method that will determine the model order should be used. One approach is to fit AR processes of progressively higher order, to calculate the squared error for each value of model order (M), and to plot this against model order. It may then be possible to see the value of M where the curve flattens out and the addition of extra parameters gives

little improvement in fit. Another approach based upon the principals of prediction is that to increase the model order until the residual process becomes a white noise.

Other criteria have been developed that are based upon concepts in mathematical statistics [9]. The first one is the final prediction error (FPE),

$$\text{FPE} = P\frac{N+M+1}{N-M-1} \tag{3.3a}$$

Where P, N and M are error, number of samples and model order respectively.

The fractional portion of FPE increases with M and accounts for the inaccuracies in estimating the parameters. The other criterion is called Akaike's information criterion (AIC). It is:

$$\text{AIC} = N\ln P^2 + 2P \tag{3.3b}$$

The first criterion tends to have a minimum at values of M that are less than the model order and the second one tends to overestimate model order.

The above criteria were calculated for electrocardiogram signal and the results were plotted in Figure 2. As shown in Figure 2(a), the error decreases but there is no definitive slope change. The largest decrease occurs from order 1 to 2 and the error does not seem to decrease significantly with orders greater than 11. For FPE (Figure 2(b)) and AIC (Figure 2(c)) plots, the error does not decrease much with orders greater than 11. Thus, the order can be approximately 10. The Levinson-Durbin algorithm was used to calculate the AR parameters with order 10 for heart pulse. These parameters were used as features.

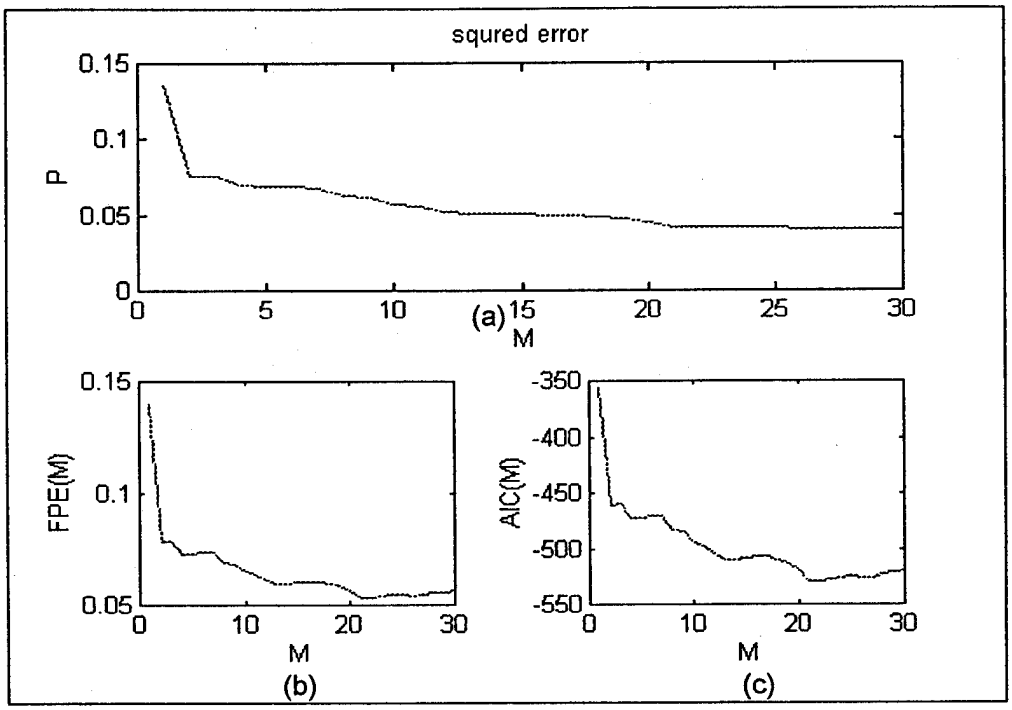


Figure 2. The different criteria for heart pulse versus model order (M): (a) error; (b) FPE; (c) AIC.

3.4 Cross-covariance and cross-correlation functions

In general, it may be necessary to study the interactions between two processes with possibly different scales of measurement or different variances. In polygraph where time series data are generated from more than one channel at a time, features like cross-correlation which contain information about relationships between the channels are extracted. The cross covariance(C_{xy}) and cross correlation function (r_{xy}) are defined as following:

$$C_{xy}(k) = \frac{\sum_{k=1}^{N-1}(X(n)-m_x)(Y(n+k)-m_y)}{N} \qquad [k = 0,1,....(N-1)] \quad (3.4a)$$

$$r_{xy} = Cxy / \sqrt{[Cxx(0)Cyy(0)]} \qquad (3.4b)$$

where $m_x = \sum_{k=1}^{N} \frac{X(n)}{N}$ $m_y = \sum_{k=1}^{N} \frac{Y(n)}{N}$ (3.4c)

$Cxx(0)$ and $Cyy(0)$ are the variances of observations on X and Y respectively.

This estimate is asymptotically unbiased. However, the variance of the estimate depends on the auto correlation functions of the two components. Therefore, for moderately large values of N it is possible for two series, which are actually uncorrelated, to give rise to large cross-correlation coefficients which are actually spurious. Thus, both series should first be filtered to convert them to white noise before computing the cross-correlation function [8].

In order to determine the relationship between the upper respiratory and heart rate, the cross correlation between them was calculated. Figure 3 shows the cross correlation between heart pulse and upper respiratory for a control and a relevant question for two different deceptive and non deceptive cases.

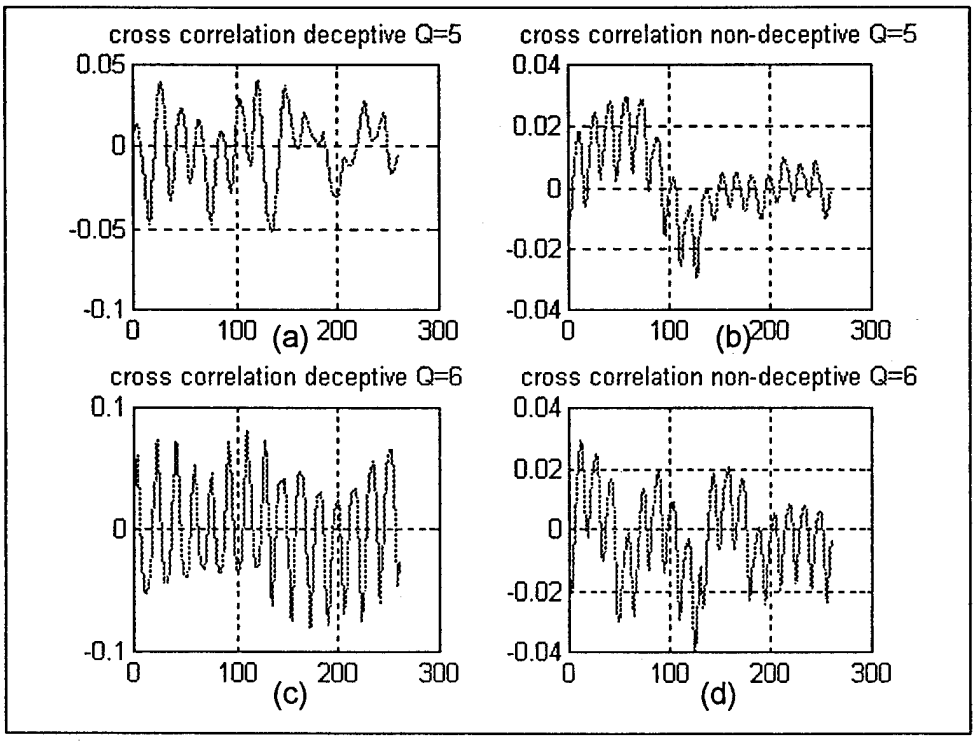


Figure 3. Cross correlation between upper respiratory and heart pulse before modeling. (a) and (b) 90 seconds after relevant question 5. (b) and (c) 90 seconds after control question 6.

3.5 Whitening filter

For a given process {x(n)}, the innovation process {v(n)} is defined as a white noise process such that {v(n)} can be determined from the signal {x(n)} by the whitening filter. The innovations representation of a random process is a powerful analytic tool. The innovation process makes the interpretation of the original process simpler than the original signal. Yet both processes contain the same statistical information. In other words, there is no loss of information as a result of the transformation.

As stated in section 3.4, it is possible for two series, which are actually uncorrelated, to give rise to large cross-correlation coefficients which are actually spurious. Thus, the series should first be filtered to convert them to white noise before computing the cross-correlation function. The AR parameters were used to design the whitening filter. Then, the heart pulse signal was filtered to convert it to white noise.

When the time series is white noise and purely random, the neighboring points of the ACF are uncorrelated. In order to compare the whitening filter output and the theoretical white noise, both the output of the whitening filter and its auto correlation for electrocardiogram were plotted in Figure 4. It is seen that the auto correlation shows high correlation for lag zero (k=175) and small correlation for other lags as it expected.

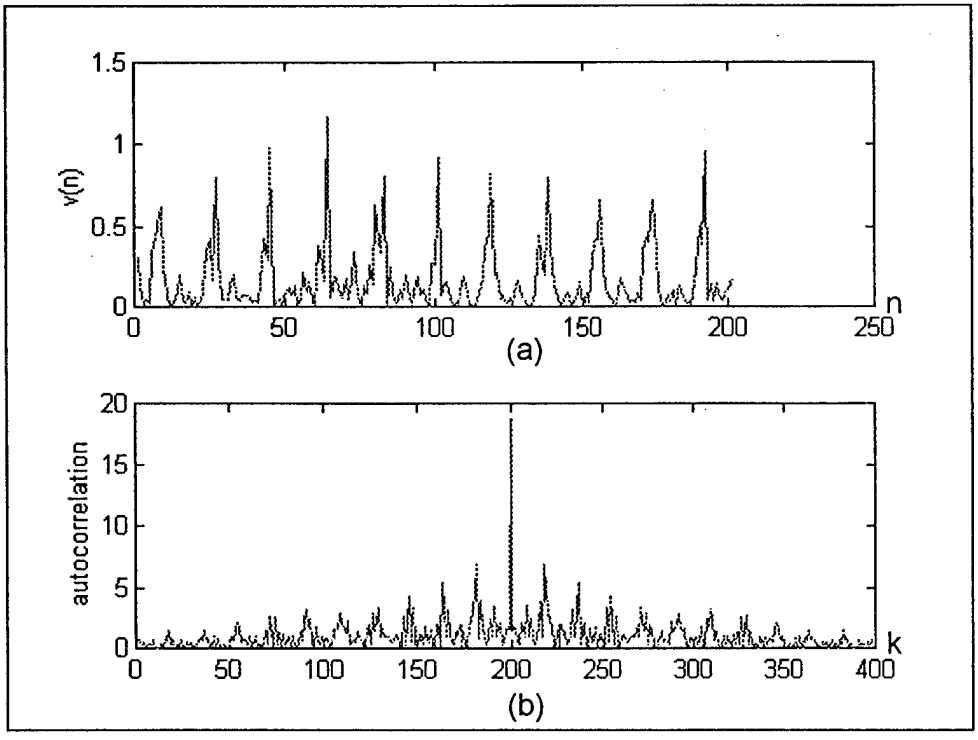


Figure 4. Plots of (a) white noise (output of the whitening filter); (b) auto correlation of the white noise.

The heart pulse and its innovation process (pre whitening filter output) contain the same information. The results of cross-correlation between upper respiratory and heart rate signals after pre whitening are shown in figure 5. It can be seen that the cross-correlation after modeling is similar to the cross correlation before modeling (Figure 2) with less spurious peaks. The maximum and minimum value of cross correlation and their lags were considered as potential features in correlation domain. As presented in figure 5 (b), heart pulse and upper respiratory channels are positively correlated after the 30 to 90 lags (1-3 seconds) and are negatively correlated after 130 lags (4.3 seconds).

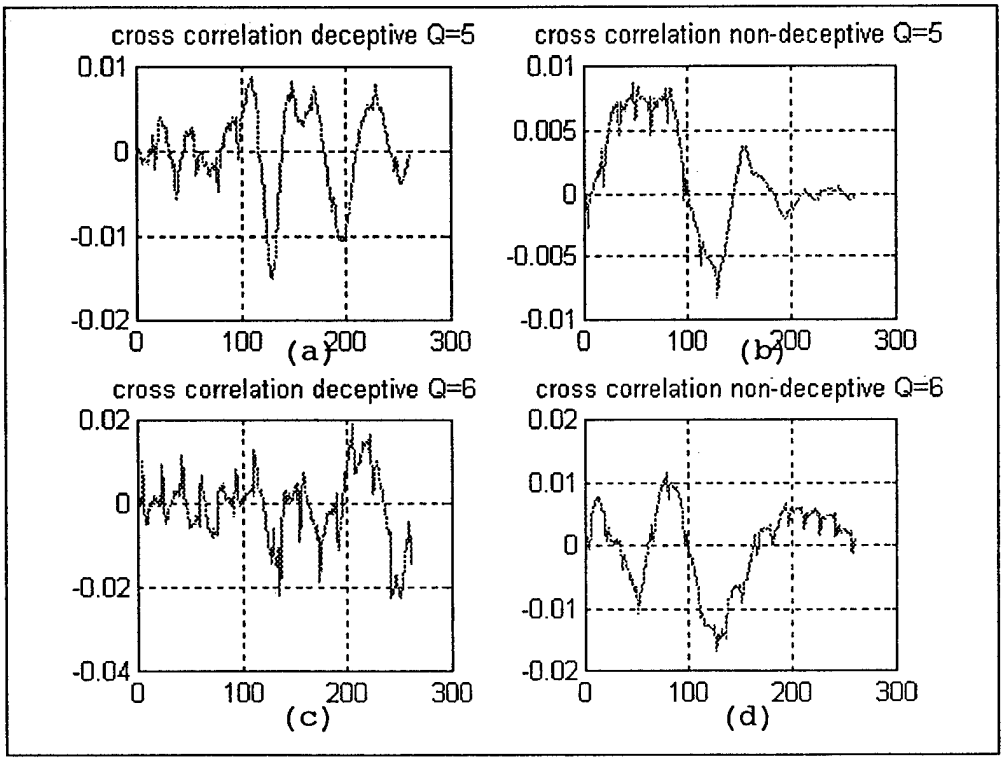


Figure 5. Cross correlation between heart pulse and upper respiratory after modeling for (a) and (b) 90 seconds after relevant question 5. (b) and (c) 90 seconds after control question 6.

3.6 Spectral Analysis

In this section the frequency properties of the polygraph signals such as power spectrum and cross spectral density are analyzed. The cross-correlation and cross spectral density are the tools for examining the relationships between two signals in the time and frequency domains respectively. The power spectrum shows how the variance of the signal is distributed with frequency. The total area underneath the spectrum curve is equal to the variance of the signal. A peak in the spectrum indicates an important contribution to the variance at different frequencies.

The estimated spectrum for different channels were plotted on linear scale in Figure 6 and on logarithmic scale in Figure 7. For spectrum showing large variations in power, a logarithmic scale makes it possible to show more detail over a wide range. However, this exaggerates the visual effects of variations where the spectrum is small. It is often easier to interpret the spectrum plotted on a linear scale than logarithmic scale.

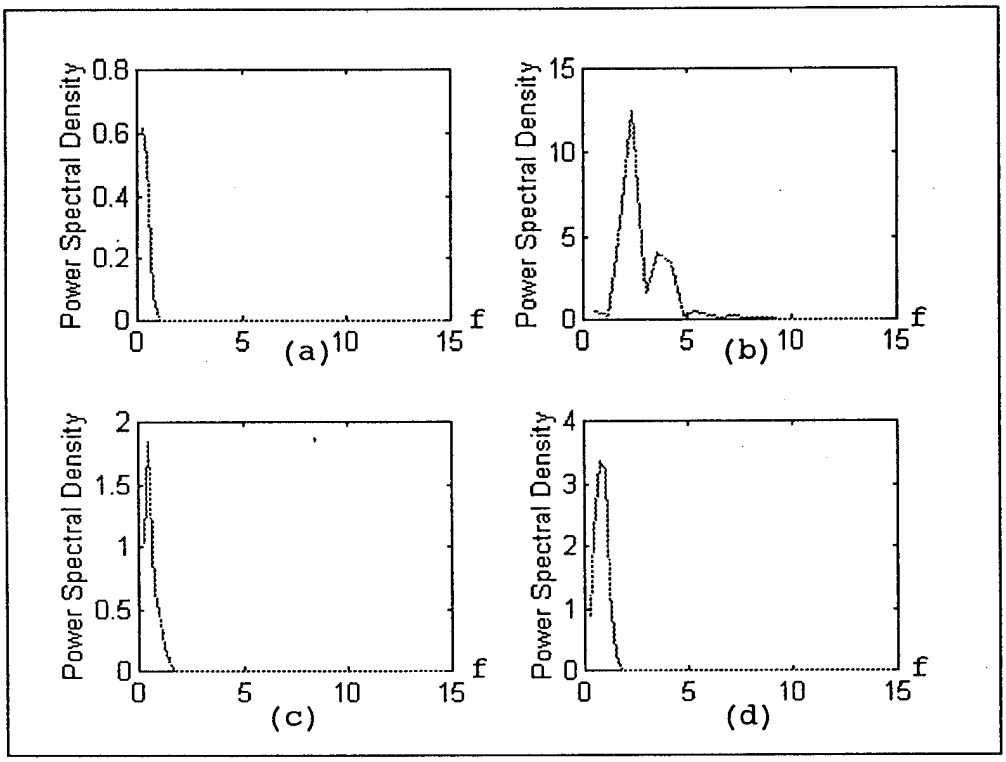


Figure 6. Frequency contents of four polygraph signals on linear scale. (a) GSR for 480 samples, (b) heart pulse for 200 samples, (c) and (d) lower and upper respiratory for 480 samples.

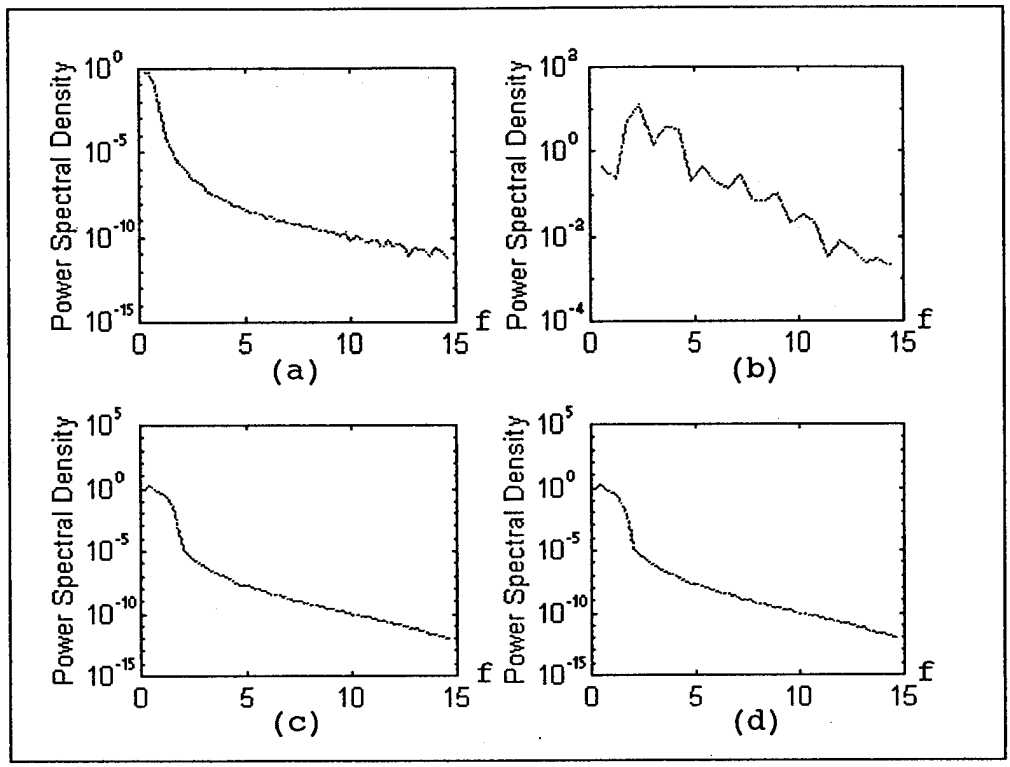


Figure 7. Frequency contents of four polygraph signals on logarithmic scale. (a) GSR for 480 samples, (b) heart pulse for 200 samples, (c) and (d) lower and upper respiratory for 480 samples.

Figure 7 shows for GSR the variance is concentrated at low frequencies indicating a trend or non-stationary behavior. The spectrum for heart pulse signal shows the presence of harmonics with a large peak at fundamental frequency of $f = 2$ Hz and related peaks at 2f, 3f, ...These multiples of the fundamental indicate the non sinusoidal character of the main cyclical component.

The correlation between two signals can be described in the frequency domain by their cross amplitude, phase spectra or the squared coherency. The coherency measures the linear correlation between the two components of the two channels at frequency f . The closer the coherency is to one, the more closely related are the two signals at frequency f.

The MATLAB function *spectrum.m* finds the cross-spectrum and coherency between upper respiratory and electrocardiogram and are shown in Figure 8. Their cross spectrum shows a large peak at $f = 2$ Hz. Maximum cross spectral density and the magnitude of cross spectral density and coherency at fundamental frequency and the second harmonic were considered as features in frequency domain.

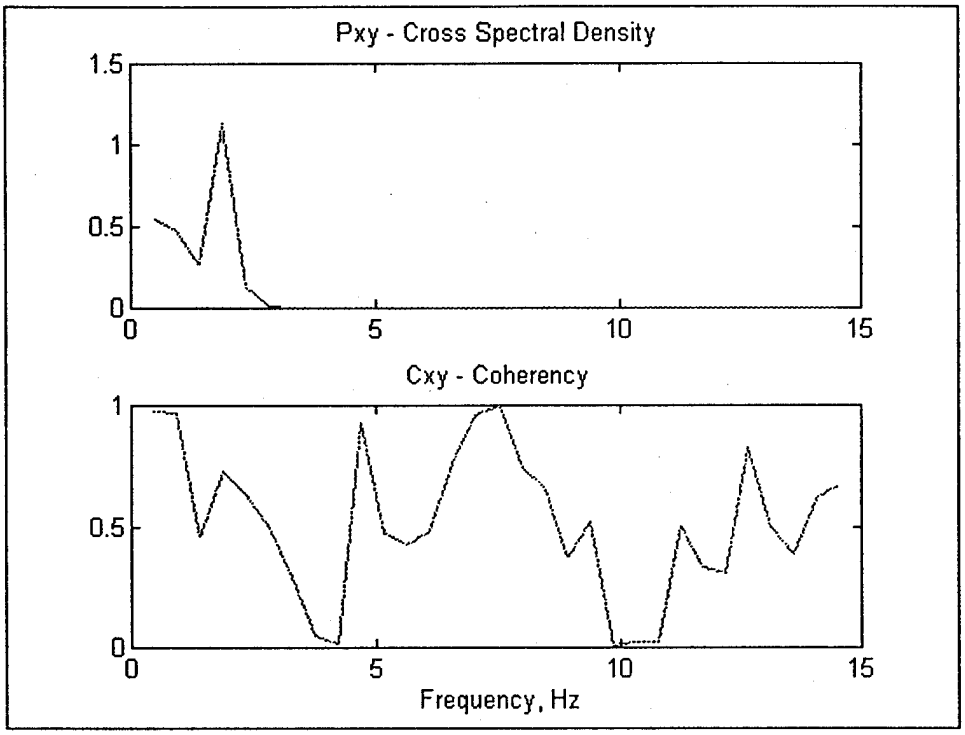


Figure 8. Plots of coherency and cross spectral density between heart pulse and upper respitor signals.

3.7 Integrated spectral distance

This section describes how to obtain a feature in the frequency domain called integrated spectral difference. This feature was introduced by Martin and Pounds [12]. Other features are calculated separately for each control, relevant and irrelevant questions. The integrated spectral distance is calculated in a different way than the other features. This feature is calculated by taking the difference between the cumulative values of the power spectral density for each relevant and its closest control question. The integrated spectral distance measures the distance between a control and a relevant question directly. Figure 9 shows the cumulative spectral density for a control and a relevant question. The maximum, the frequency where this maximum happens and the area underneath were considered as features.

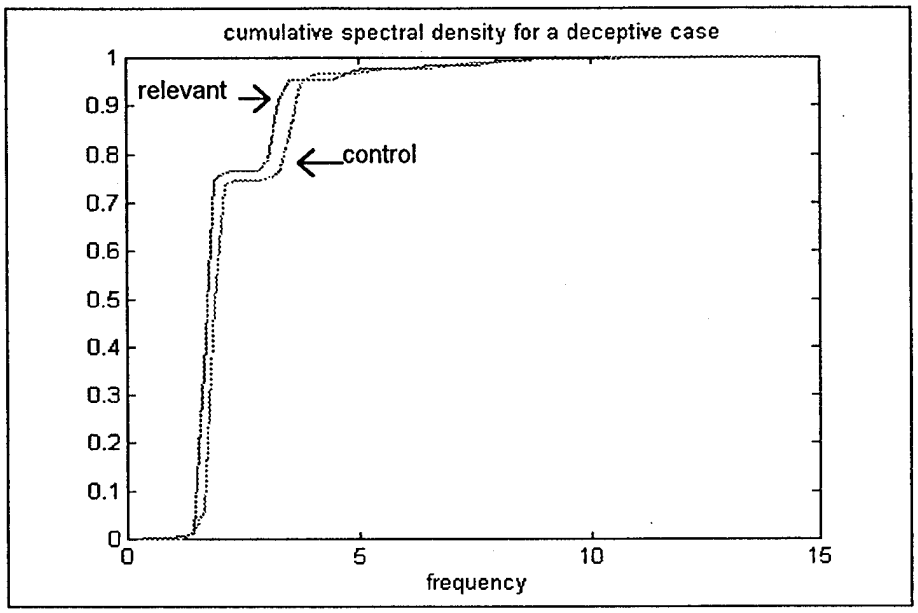


Figure 9. Cumulative integrated spectral density for a control question and relevant question of the heart pulse signal.

3.8 Frequency and Correlation Domain Features

Table 1 summarizes the frequency and correlation features explained in the above sections.

Feature	Channel
Maximum cross correlation	between 2 & 6
Lag of maximum cross correlation	between 2 & 6
Minimum cross correlation	between 2 & 6
Lag of minimum cross correlation	between 2 & 6
Spectral value at fundamental frequency	2
Spectral value at fundamental frequency	6
Spectral value at (fundamental frequency of channel 2) *2	2
Spectral value at (fundamental frequency of channel 6) *2	6
Maximum cross spectral density	between 2 & 6
Coherency at fundamental frequency of channel 2	between 2 & 6
Coherency (at fundamental frequency of channel 2)*2	between 2 & 6
Fundamental frequency	2
Fundamental frequency	5
Maximum or minimum integrated spectral difference	1
Frequency of the maximum integrated spectral difference	1
Area underneath integrated spectral difference	1
maximum or minimum integrated spectral difference	2
Frequency of the maximum integrated spectral difference	2
Area underneath integrated spectral difference	2
Auto regressive parameter	2

Table 1. Frequency and correlation domain features.

4 Feature extraction

4.1 Preprocessing

This chapter explains the steps taken in feature extraction algorithm. In polygraph tests, four physiological responses are measured. These responses are: upper respiratory, lower respiratory, galvanic skin response (GSR) and electrocardiogram. These four polygraph responses are processed into six channels. A low frequency electrocardiogram channel is produced by lowpass filtering the electrocardiogram channel. A high frequency electrocardiogram channel is produced by highpass filtering it. The high frequency electrocardiogram, called heart pulse, the low frequency electrocardiogram, called blood volume and derivative of the low frequency electrocardiogram are used instead of one electrocardiogram channel. To eliminate the noise and any trend, all the signals are filtered and detrended. For more information about the filtering and detrending refer to Jacobs [10].

4.2 Feature Selection

Many of the time domain features were selected based on the examiners' suggestions. However, many of the standard statistical features were also considered as potential features. For more information about time domain features refer to Jacobs [10]. The selected features and the channels which they were extracted from are listed below.

Features	Channel
1) Mean	1 ,2 , 3, 4, 5 , 6
2) Standard deviation	1 ,2 , 3, 4, 5 , 6
3) Minimum	1 ,2 , 3, 4, 5 , 6
4) Maximum	1 ,2 , 3, 4, 5 , 6
5) Curve length	1 ,2 , 3, 4, 5 , 6
6) Mean of derivative	1 ,2 , 3, 4, 5 , 6
7) Median of derivative	1 ,2 , 3, 4, 5 , 6
8) Average amplitude of peaks	2 , 5 , 6
9) Minimum amplitude of peaks	2 , 5 , 6
10) Derivative of amplitudes of peaks	2 , 5 , 6
11) Number of peaks	2 , 5 , 6
12) Minimum subtracted from maximum	1 ,2 , 3, 4, 5 , 6
13) Inhalation/exhalation	5 , 6
14) ratio of inhalation/exhalation before and after a question is asked	5 , 6
15) Fundamental frequency	2, 5
16) Maximum cross correlation	between 2 and 6
17) Lag of maximum cross correlation	between 2 and 6
18) Minimum cross correlation	between 2 and 6
19) Lag of minimum cross correlation	between 2 and 6
20) Spectral value at fundamental frequency	between 2 and 6
21) Spectral value at second harmonic	between 2 and 6
22) Maximum cross spectral density	between 2 and 6
23) Coherency at fundamental frequency	between 2 and 6
24) Coherency at second harmonic	between 2 and 6
25) Autoregresive parameters(AR)	2
26) Maximum or minimum integrated spectral difference (ISD)	1, 2
27) Frequency of maximum ISD	1, 2
28) Area under ISD	1, 2

4.3 Feature Extraction Algorithm

All features are extracted for 10 relevant, irrelevant and control questions except features 26, 27 and 28 that are extracted for each relevant and its closest control question. The program called fextract.m extracts all the basic features for each question on each chart for about 18 non-deceptive and 51 deceptive cases. Due to the small number of non-deceptive cases, each chart for a subject was used as a separate case. By doing this 50 non-deceptive and 150 deceptive files were created.

The test format used in this project is MGQT format. It is a type of control question test in which relevant, irrelevant and control questions are asked in a specific order. Each polygraph test is made of three and in very rare cases four charts for each case. The order in which the questions are asked is changed in the third and fourth charts and sometimes in the second chart. The feature extraction routine needs to have the control, relevant and irrelevant questions labeled. Therefore, for each polygraph chart a complementary chart called question file was created which contains a matrix called Q. The first row of this matrix contains the relevant, the second row the irrelevant and the third row the control questions respectively.

Fragments of each signal are selected before features are extracted. These fragments are shown in Table 2. Start and end points given in the table refer to the time elapsed after the question is asked. A vector of features for each file is created by the program feature.m which is called by fextract.m program. The program first executes all of the processing routines and then extracts the features for each question in the file. The features are extracted for the appropriate time segment (see Table 2) of six channels for each polygraph file. The time segment is created by taking a sample of time series starting several seconds after a question is asked and continuing for a number of seconds.

Channel description	Channel	Start	End
Galvanic Skin conductivity(GSR)	1	2 sec.	14 sec.
High frequency electrocardiogram	2	2 sec.	9 sec.
Low frequency electrocardiogram (LC)	3	2 sec.	18 sec.
Derivative of low frequency electrocardiogram (DLC)	4	0 sec.	8 sec.
Lower Respiratory (LR)	5	2 sec.	18 sec.
Upper Respiratory (UR)	6	2 sec.	18 sec.

Table 2. Time fragment used in feature extraction

The feature extraction algorithm provides a 960 dimensional vector for each file. The features were extracted for the 150 deceptive and 50 non deceptive files and saved in a 960 by 200 matrix called " M". In order to classify subjects using the difference between control and relevant responses, and to make the feature vector smaller, the features were combined according to the following method: for each feature *i* except features 26, 27,28 from each subject *j* compute:

1) The average control responses *AvCij*
2) The average relevant responses *AvRij*
3) The maximum and minimum control responses *MaxCij* and *MinCij*
4) The maximum and minimum relevant responses *MaxRij* and *MinRij*

The feature vector components for feature i are then:

$$1) Fij(1) = AvRij - AvCij$$

$$2) Fij(2) = \frac{AvRij - AvCij}{AvRij + AvCij}$$

$$3) Fij(3) = MaxRij - MaxCij$$

$$4) Fij(4) = MinRij - MinCij$$

$$5) Fij(5) = MaxRij - MinCij$$

$$6) Fij(6) = MinRij - MaxCij$$

$$7) Fij(7) = \frac{MaxRij}{MaxCij}$$

For features 26, 27, 28 from each subject *j* compute:

1) The average of relevant-control responses *Av(RC(ij)*
2) The maximum of relevant-control responses *Max(RC(ij)*
3) The minimum of relevant-control responses *Min(RC(ij)*

The feature vector components for feature *i* are then:

$$1) F_{ij}(1) = Av(RC(_{ij}))$$

$$2) F_{ij}(2) = Max(RC(_{ij}))$$

$$3) F_{ij}(3) = Min(RC(_{ij}))$$

The above procedure is executed by program called *procesf.m* which creates a 669 by 200 dimensional matrix called "F". In order to run the classifier program, the matrix F was divided into three 100 (50 deceptive and 50 non-deceptive) sets of matrices called set1, set2 and set3. These sets are made of 50 non-deceptive cases common in all three sets and three 50 different deceptive sets, called deceptive 1, deceptive 2 and deceptive 3 respectively. The list of the files used in the set1, set2 and set3 are shown in Table 3 in Appendix A.

5 Results

5.1 Frequency Domain Clustering

Classifier is the final stage in a pattern recognition system. The classifier assigns each input to one of the classes. The classifier could be designed after studying the distribution of samples in each class. The KNN classifier was used in this study because of the following:

1) The uncertainty about the shape of deceptive and non deceptive clusters and their sample distributions.
2) The possibility that the samples for one class cluster around more than one point in space.

The frequency domain features did not create a separate distribution of samples for deceptive and non deceptive classes. However, the combination of frequency and time domain features resulted in more distinct clusters. Figure 10 and 11 show the examples of sample distribution (clustering) for non deceptive (x) and deceptive (+) classes.

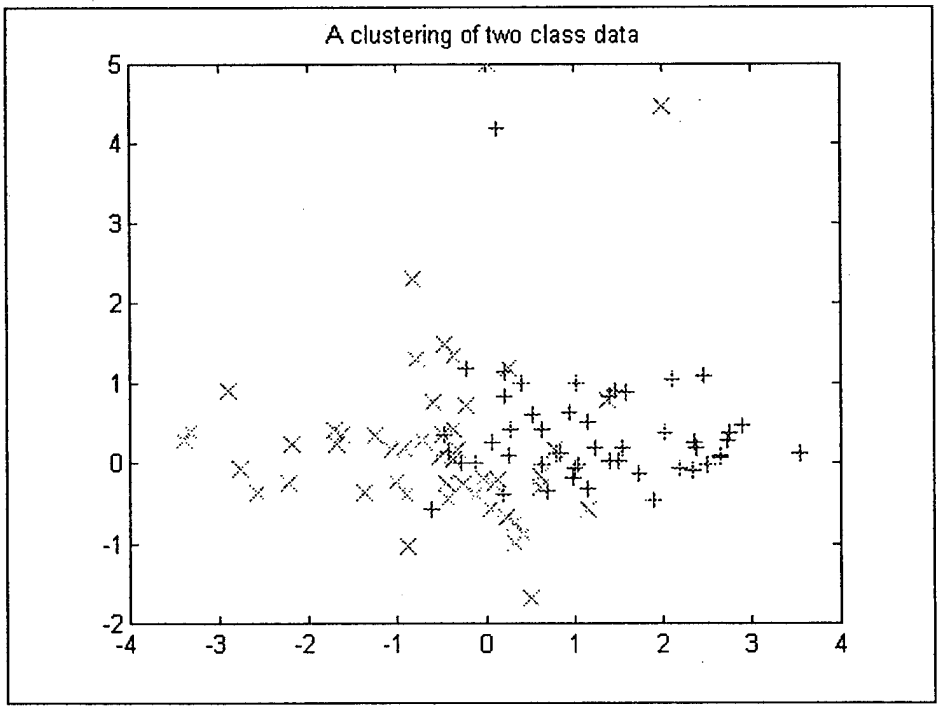


Figure 10. Plot of maximum of GSR versus maximum of Upper Respiratory.

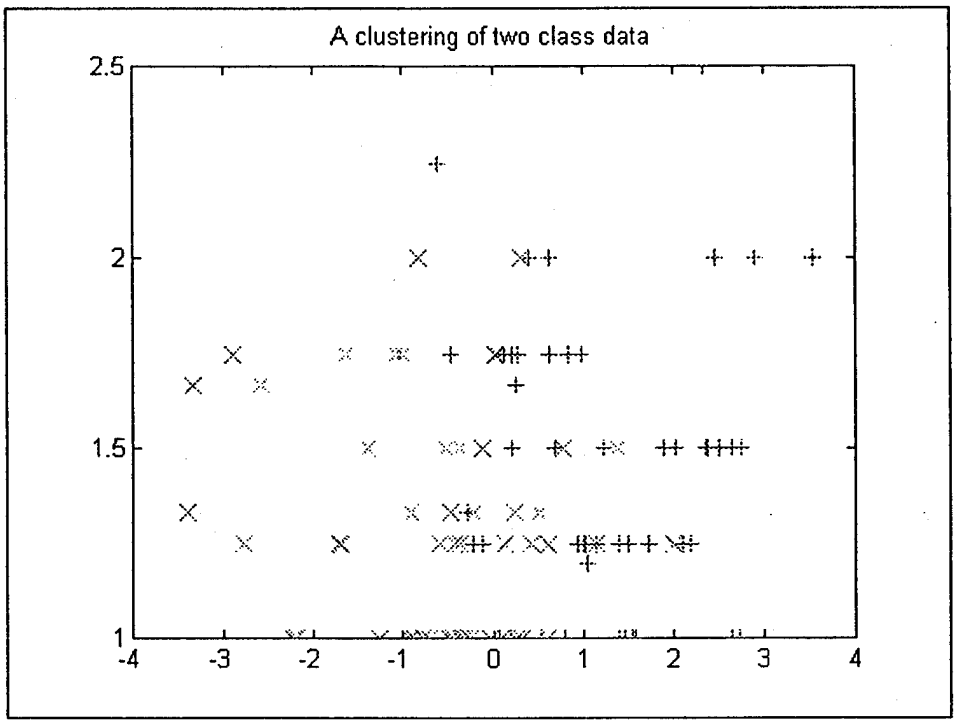


Figure 11. Plot of maximum of GSR versus frequency of maximum integrated spectral difference of GSR.

5.2 Discussion

The 669 features are more than can be used by any classification techniques. Thus, the classification program and the scatter measurement program were run for each feature in each set individually. The results of the first experiment were examined and compared to determine the features which were the best discriminators between deceptive and non-deceptive subjects. After comparing the results, the 30 features with the highest accuracy rate and common in all three sets were selected. These best features were listed in Table 3.

The second experiment used the combination of two features out of the best 30 features. The results for the best 30 features were examined for each set separately. The set3 always had a better performance than the other two sets. However, in order to be consistent, the best features common in all three sets were selected as the 30 best features. More features were added for combination of three and four. The results are shown in Table 4 and 5 in Appendix A.

As it was discussed before, the classifier was used to compare the effectiveness of the single features and to choose the combination of the best features. Changing the classifier parameters such as K might change the results of the classification. However, it is not practical to change all parameters at the same time. Therefore, the classifier was used with the fixed parameters of K=5 and m=2. After selecting the final feature set, theses parameters were changed to find the best classification.

No	feature	Description	Channel	Method
1	10mean	mean	GSR	1
2	10curve	curve length	GSR	2
3	10med_dif	median of the derivative	GSR	1
4	10max_min	minimum subtracted from the maximum	GSR	2
5	10max	maximum of the signal	GSR	1
6	10mdif	mean of derivative	GSR	3
7	20curve	curve length	Heart pulse	1
8	20ampcard	amplitude of the peaks	Heart pulse	1
9	20max_min	minimum subtracted from the maximum	Heart pulse	4
10	20max	maximum of the signal	Heart pulse	4
11	20min	minimum of the signal	Heart pulse	1
12	30med_dif	median of the derivative	Blood pressure	3
13	30max	maximum of the signal	Blood pressure	1
14	40mean	mean	Derivative of Blood pressure	1
15	40max	maximum of the signal	Derivative of Blood pressure	1
16	50curve	curve length	Lower Respiratory	6
17	50ampr	amplitude of the peaks	Lower Respiratory	2
18	50peaknumr	number of the peaks	Lower Respiratory	5
19	50ie	inhalation divided by exhalation	Lower Respiratory	5
20	50max_min	minimum subtracted from the maximum	Lower Respiratory	2
21	50max	maximum of the signal	Lower Respiratory	6
22	60max_min	minimum subtracted from the maximum	Upper Respiratory	2
23	60max	maximum	Upper Respiratory	3
24	10std	standard deviation	GSR	2
25	20std	standard deviation	Heart pulse	1
26	50std	standard deviation	Upper Respiratory	6
27	20armod1	auto regressive parameter	Heart pulse	7
28	26psdcoh1	max cross spectral density	Heart pulse, Lower Respiratory	1
29	10isd1	frequency of maximum integrated spectral difference of control-relevant pair	GSR	1*
30	20isd1	area under integrated spectral difference	Heart pulse	3*

Methods: 1=Difference of Averages, 2=Normalized Average, 3=Max-Max, 4=Min-Min,

5=Max-Min, 6=Min-Max, 7=Max/Min , 1*=Average of relevant-control pairs, 3*=Max of relevant-control pair.

Table 3. 30 best selected Features

Conclusion

The classification results improved consistently by increasing the number of features. The best features are {5 9 21 23} and {5 21 23 29}with 81 and 80 percent correct classification respectively. These features are maximum of GSR(5), difference between maximum and minimum of heart pulse(9), maximum of lower respiratory(21), maximum of upper respiratory(23) and frequency of maximum integrated spectral difference of control-relevant pair for GSR(29).

The best features are simple and obvious features such as maximum and minimum of the polygraph signals. In other words, the features that an examiner can see are the best discriminators between deceptive and non deceptive.

It is important to notice that the best features are the combination of features from all 4 different GSR, heart pulse, lower and upper respiratory. As expected, each subject shows reaction to different channels. Therefore, the combination of all channels is the best representative of deception.

Another point to notice is that the set3 has better classification results than the other two sets. For example, the features {9 14 19 24} and {5 21 23 29}show 87.4 and 86.6 percent correct classification for set3. The data in set3 is made of 50 non deceptive common in all three sets and 50 deceptive cases. This set of deceptive cases, called deceptive 3, are the Acxiton files listed in Table 3 in Appendix A. It is possible that there is some characteristic in these deceptive files that results in better classification.

As stated before, due to the small number of non-deceptive cases available, each chart for a subject was used as a separate case. After classifying the charts, the charts for each case were combined in a way that each case was assigned to the class that the majority of the charts belong to. Using this method, the classification results improved from 81 percent to 85.6 percent for set1 and set2 and from 87 percent to 91 percent for set3. The final result is included in appendix A.

References

[1] Dale E. Olsen, et. al., "Recent developments in polygraph testing: A research review and evaluation - A technical memorandum, " Washington, DC: US Government Printing Office 1983.

[2] John E. Reid and Fred E. Inbau, Truth and Deception: The Polygraph (" Lie Detector ") Technique, The Williams & Wilkins Company, Baltimore, Md., 1966

[3] Michael H. Capps and Norman Ansley, "Numerical Scoring of Polygraph Charts: What Examiners Really Do", Polygraph, 1992, 21, 264-320

[4] Personal communication with Richard Petty (polygraph examiner), June 1993

[5] J.M.Keller, M.R. Gray and J.A. Givens, "A fuzzy K nearest neighbor algorithm", IEEE Trans. on syst.Man Cybernetics, vol SMC-15, No.14

[6] J.C. Bezdek and Siew K.Chuan, "Generalized K nearest neighbor rules, Fuzzy sets and System vol 18(1986).

[7] Rabiner and Schafer, "Digital Processing of Speech Signals", p141.

[8] Jenkins and Watts, 1968, p. 340.

[9] Richard Shiavi, "Introduction to Applied Statistical Signal Analysis", P357.

[10] Eric Jacobs ,"Time Domain Features of Polygraph Data", Masters Project Report, San Jose State University, Fall 1993.

[11] Shahab Layeghi, "Pattern Recognition of the Polygraph Using Fuzzy Set Theory",Masters Project report, San Jose State University, Fall 1993.

[12] R. Douglas Martin, Ph.D. and Christopher B. Pounds, Polygraph Reliability, the Department of Statistics University of Washington Seattle, Washington 98195 October1,1991- September 30, 1992.

Appendices

Appendix A

Tables

FILE NAME	FUNDAMENTAL FREQUENCY (Hz) CHANNEL : Heart pulse, WINDOW: 120 S			
QQAV53P6.021	relevant = 1.3636	1.3636	1.3636	1.4286
	control = 1.2500	1.5000		
QQAV53P6.031	relevant = 1.5000	1.3636	1.3043	1.3636
	control = 1.4286	1.3636	1.3636	1.4286
QQBQ4SHI.011	relevant = 2	2	2	2
	control = 2	2		
QQBQ4SHI.021	relevant = 1.7647	1.7647	1.7647	1.8750
	control = 1.8750	1.76		
QQBQ4SHI.031	relevant = 1.7647	1.7647	1.7647	1.7647
	control = 0.8571	1.7647	1.7647	1.6667
QQBSS7WT.011	relevant = 1.5000	1.5000	1.5000	1.3636
	control = 1.5789	1.4286		
QQBSS7WT.021	relevant = 1.5000	1.4286	1.4286	1.4286
	control = 1.5000	1.4286		
QQBSS7WT.031	relevant = 1.4286	1.5000	1.4286	1.3636
	control = 1.4286	1.5000	1.4286	1.5000

Table 1. Fundamental frequency for non-deceptive files for 120 seconds for heart pulse.

FILE NAME	FUNDAMENTAL FREQUENCY(Hz) CHANNEL : CARDIO, WINDOW: 120 S				
QQ9SOW8L.021	relevant =	1.7647	1.6667	1.5789	1.6667
	control =	1.5789	1.5789		
QQ9SOW8L.031	relevant=	1.5789	1.5789	1.6667	1.6667
	control =	1.8750	1.6667	1.7647	1.5789
QQ9SQIK9.011	relevant =	1.5789	1.5000	1.5000	1.5789
	control =	1.5789	1.5000		
QQ9SQIK9.021	relevant =	1.3043	1.5789	1.5789	1.4286
	control =	1.5789	1.5789		
QQ9SQIK9.031	relevant =	1.5000	1.5000	1.6667	
	control =	1.4286	1.2000	1.5789	1.5789
QQ9W0B9F.011	relevant =	1.5000	1.4286	1.5000	1.5000
	control =	1.4286	1.5789		
QQ9W0B9F.031	relevant=	1.4286	1.5000	1.4286	1.4286
	control =	1.5000	1.4286		
QQ9W0B9F.041	relevant =	1.4286	1.3636	1.4286	1.5000
	control =	1.4286	1.3636		
QQ9U4FMU.011	relevant =	1.5789	1.6667	1.6667	1.6667
	control =	1.6667	1.5789		

Table 2. Fundamental frequency for deceptive files for 120 seconds for heart pulse.

Non deceptive	Deceptive 1	Deceptive 2	Deceptive 3
QQ8R9OIO.011	QQ4Q1O83.011	QQ7LX5Q0.021	QQ8RAJ0C.011
QQ8R9OIO.021	QQ4Q1O83.021	QQ7LX5Q0.031	QQ8RAJ0C.021
QQ8R9OIO.031	QQ4Q1O83.031	QQ7MN2Y0.011	QQ8RAJ0C.031
QQ95LU1T.011	QQ4Q3MDC.011	QQ7MN2Y0.021	QQ9EUKVT.011
QQ95LU1T.021	QQ4Q3MDC.021	QQ7MN2Y0.031	QQ9EUKVT.021
QQ95LU1T.031	QQ4Q3MDC.031	QQ7TC5UF.011	QQ9EUKVT.031
QQAURNUS.021	QQ51DE36.011	QQ7TC5UF.021	QQ9IOOXO.021
QQAURNUS.031	QQ51DE36.021	QQ7TC5UF.031	QQ9IOOXO.041
QQAV53P6.011	QQ51DE36.041	QQ7TQVER.011	QQ9SOW8L.011
QQAV53P6.021	QQ6RQGH6.011	QQ7TQVER.021	QQ9SOW8L.021
QQAV53P6.031	QQ6RQGH6.021	QQ7TQVER.031	QQ9SOW8L.031
QQBQ4SHI.011	QQ6RQGH6.031	QQ7TVADC.011	QQ9SQIK9.011
QQBQ4SHI.021	QQ6RQGH6.041	QQ7TVADC.021	QQ9SQIK9.021
QQBQ4SHI.031	QQ6T711O.011	QQ7TVADC.031	QQ9SQIK9.031
QQBSS7WT.011	QQ6T711O.021	QQ7U2T4R.011	QQ9W0B9F.011
QQBSS7WT.021	QQ6T711O.031	QQ7U2T4R.021	QQ9W0B9F.031
QQBSS7WT.031	QQ6Z59IG.011	QQ7U2T4R.031	QQ9W0B9F.041
QQ7OXM60.021	QQ6Z59IG.021	QQ7YP7QU.011	QQ9U4FMU.011
QQ7RH0RO.011	QQ6Z59IG.031	QQ7YP7QU.021	QQ9U4FMU.021
QQ7RH0RO.021	QQ7PP9B9.011	QQ7YP7QU.031	QQ9U4FMU.031
QQ7RH0RO.031	QQ7PP9B9.021	QQ7YZOJ3.011	QQ9Y_SVF.011
QQ7R51P9.011	QQ7PP9B9.031	QQ7YZOJ3.021	QQ9Y_SVF.021
QQ7R51P9.021	QQ7PDU1X.011	QQ7YZOJ3.031	QQ9Y_SVF.031
QQ7R51P9.031	QQ7PDU1X.021	QQ8_0DPT.011	QQ9YH3QF.011
QQ9TDSP3.011	QQ7PDU1X.031	QQ8_0DPT.021	QQ9YH3QF.021
QQ9TDSP3.021	QQ7_PIPF.011	QQ8_0DPT.031	QQ9YH3QF.031
QQ9TDSP3.031	QQ7_PIPF.021	QQ8_0DPT.041	QQA2TT4C.011
QQA8OWOI.011	QQ7_PIPF.031	QQ8_2UQ9.011	QQA2TT4C.021
QQA8OWOI.021	QQ7_JT70.011	QQ8_2UQ9.021	QQA2TT4C.031
QQA8OWOI.031	QQ7_JT70.021	QQ8_2UQ9.031	QQA3HIRX.011
QQBT22O6.011	QQ7_JT70.031	QQ800IG6.011	QQA3HIRX.021
QQBT22O6.021	QQ738DYX.011	QQ800IG6.021	QQA3HIRX.031
QQBT22O6.031	QQ738DYX.021	QQ800IG6.031	QQA32UTF.011
QQBO9O_9.011	QQ738DYX.031	QQ82OIU9.011	QQA32UTF.021
QQBO9O_9.021	QQ75ULP9.011	QQ82OIU9.021	QQA32UTF.031
QQBO9O_9.031	QQ75ULP9.021	QQ82OIU9.031	QQA6U_IF.011
QQBC7PP6.011	QQ75ULP9.031	QQ82SUTX.011	QQA6U_IF.031
QQBC7PP6.021	QQ79_EYF.011	QQ82SUTX.021	QQA6U_IF.041
QQBC7PP6.031	QQ79_EYF.021	QQ82SUTX.031	QQAM4E3L.011
QQCHCK_O.011	QQ79_EYF.031	QQ860ZNU.011	QQAM4E3L.021
QQCHCK_O.021	QQ7BGDML.011	QQ860ZNU.021	QQAM4E3L.031
QQCHCK_O.031	QQ7BGDML.021	QQ860ZNU.031	QQARF2_X.011
QQCDTKP0.011	QQ7BGDML.031	QQ89U_ZR.011	QQARF2_X.021
QQCDTKP0.031	QQ7ETC8I.011	QQ89U_ZR.021	QQARF2_X.031
QQCDTKP0.041	QQ7ETC8I.021	QQ89U_ZR.031	QQAWA38X.011
QQCM5Y56.011	QQ7ETC8I.031	QQ8ATU26.011	QQAWA38X.021
QQCQQT8Y.011	QQ7JAQCS.011	QQ8ATU26.021	QQAWA38X.031
QQCQQT8Y.021	QQ7JAQCS.021	QQ8ATU26.031	QQAYXZGU.011
QQCQQT8Y.031	QQ7JAQCS.031	QQ8FGMVI.011	QQAYXZGU.021
QQCQQT8Y.041	QQ7LX5Q0.011	QQ8FGMVI.021	QQAYXZGU.031

Table 3. List of files used in this experiment. 50 non-deceptive cases and 50 deceptive cases from set1, set2 and set3 are listed in column 1 through 4 respective

Set	Features	accuracy
Set1	10 21 26	79.4
	5 11 23	77.6
	5 21 23	77.4
Set2	12 20 24	79.8
	19 24 30	78.6
	5 21 23	77.4
Set3	9 19 24	85.2
	5 23 29	82.4
	5 21 23	81.2
Average	5 23 29	78.2
	5 7 23	77.6
	5 21 23	77.3

Table 4. The three best features of combination of 3 for each set and their average.

Set	Features	accuracy
Set1	5 9 21 23	81.0
	5 11 21 23	80.2
	5 21 23 29	74.4
Set2	5 14 23 29	81.0
	5 9 21 23	79.4
	5 21 23 29	79.0
Set3	9 14 19 24	87.4
	5 21 23 29	86.6
	5 21 23 9	82.5
Average	5 9 21 23	81.0
	5 21 23 29	80.0
	5 21 23 11	79.8

Table 5. The three best features of combination 4 for each set and their average.

File	Membership	Defuzzified	Result
1.0000	0.2736	0	
2.0000	0.3339	0	
3.0000	0.5397	0	0
4.0000	0.5450	0	
5.0000	0.7423	1.0000	
6.0000	0.1732	0	0
7.0000	0.8901	1.0000	
8.0000	1.0000	1.0000	1 Misclassified
9.0000	0.5376	0	
10.0000	0.1742	0	
11.0000	0.4366	0	0
12.0000	0.3458	0	
13.0000	0.5145	0	
14.0000	0.5178	0	0
15.0000	0.1016	0	
16.0000	0	0	
17.0000	0	0	0
18.0000	0.1334	0	0
19.0000	0	0	
20.0000	0	0	
21.0000	0.2923	0	0
22.0000	0	0	
23.0000	0	0	
24.0000	0.1607	0	0
25.0000	0	0	
26.0000	0.4421	0	
27.0000	1.0000	1.0000	0
28.0000	0.3307	0	
29.0000	0.0583	0	
30.0000	0.4965	0	0
31.0000	0.3505	0	
32.0000	0.1181	0	
33.0000	0.2101	0	0

Table 6. Classification of the files in Set1.

File	Membership	Defuzzified	Result
34.0000	0.5970	0	
35.0000	0	0	
36.0000	0.1193	0	0
37.0000	0.3174	0	
38.0000	0.8117	1.0000	
39.0000	0.0997	0	0
40.0000	0.1889	0	
41.0000	0.4215	0	
42.0000	0.1635	0	0
43.0000	0.6474	1.0000	
44.0000	0	0	
45.0000	0.5495	0	0
46.0000	0.1115	0	0
47.0000	0	0	
48.0000	0.3986	0	
49.0000	0	0	
50.0000	0	0	0
51.0000	0.6709	1.0000	
52.0000	1.0000	1.0000	
53.0000	0.5297	0	1
54.0000	0.7245	1.0000	
55.0000	0.9200	1.0000	
56.0000	1.0000	1.0000	1
57.0000	0.9105	1.0000	
58.0000	0.9398	1.0000	
59.0000	0.5657	0	1
60.0000	0.8968	1.0000	
61.0000	1.0000	1.0000	
62.0000	0.2793	0	
63.0000	0.1088	0	0 **Misclassified**
64.0000	0.6245	1.0000	
65.0000	0.8643	1.0000	
66.0000	0.5054	0	1

Table 6. Continued.

File	Membership	Defuzzified	Result
67.0000	0.8498	1.0000	
68.0000	0.6969	1.0000	
69.0000	0.8397	1.0000	1
70.0000	0.2901	0	
71.0000	0.8291	1.0000	
72.0000	0.3982	0	0 Misclassified
73.0000	1.0000	1.0000	
74.0000	0.2463	0	
75.0000	0.8043	1.0000	1
76.0000	0.6676	1.0000	
77.0000	1.0000	1.0000	
78.0000	1.0000	1.0000	1
79.0000	1.0000	1.0000	
80.0000	0.7538	1.0000	
81.0000	1.0000	1.0000	1
82.0000	1.0000	1.0000	
83.0000	0.8378	1.0000	
84.0000	1.0000	1.0000	1
85.0000	0.8926	1.0000	
86.0000	0.5448	0	
87.0000	0.5751	0	0 Misclassified
88.0000	0.8273	1.0000	
89.0000	0.2945	0	
90.0000	0.9110	1.0000	1
91.0000	1.0000	1.0000	
92.0000	1.0000	1.0000	
93.0000	0	0	1
94.0000	0.2887	0	
95.0000	0.2079	0	
96.0000	0.5793	0	0 Misclassified
97.0000	1.0000	1.0000	
98.0000	0.7971	1.0000	
99.0000	0.8708	1.0000	1
100.0000	1.0000	1.0000	1

Table 6. Continued.

File	Membership	Defuzzified	Result
1.0000	0.2579	0	
2.0000	0.1307	0	
3.0000	0	0	0
4.0000	0.2652	0	
5.0000	0.4345	0	
6.0000	0.1175	0	0
7.0000	1.0000	1.0000	
8.0000	0.7086	1.0000	1 **Misclassified**
9.0000	0.2856	0	
10.0000	0.2745	0	
11.0000	0.3056	0	0
12.0000	0.2720	0	
13.0000	0.5019	0	
14.0000	0.8871	1.0000	0
15.0000	0.0912	0	
16.0000	0	0	
17.0000	0	0	0
18.0000	0.8334	1.0000	1 **Misclassified**
19.0000	0	0	
20.0000	0	0	
21.0000	0.5483	0	0
22.0000	0	0	
23.0000	0	0	
24.0000	0.1535	0	0
25.0000	0.4955	0	
26.0000	0.1013	0	
27.0000	1.0000	1.0000	0
28.0000	0.3788	0	
29.0000	0.1638	0	
30.0000	0.0905	0	0
31.0000	0	0	
32.0000	0.1431	0	
33.0000	0.0937	0	0

Table 7. Classification of the files in set2.

File	Membership	Defuzzified	Result
34.0000	0	0	
35.0000	0	0	
36.0000	0.1281	0	0
37.0000	0.3690	0	
38.0000	0.5734	0	
39.0000	0.1569	0	0
40.0000	0.3659	0	
41.0000	0.4124	0	
42.0000	0.1704	0	0
43.0000	0.4251	0	
44.0000	0.0664	0	
45.0000	0.5356	0	0
46.0000	0.5084	0	0
47.0000	0.1735	0	
48.0000	0.7512	1.0000	
49.0000	0.5115	0	
50.0000	0.0976	0	0
51.0000	0.6361	1.0000	
52.0000	0.8482	1.0000	1
53.0000	0.3471	0	
54.0000	0.8822	1.0000	
55.0000	1.0000	1.0000	1
56.0000	1.0000	1.0000	
57.0000	1.0000	1.0000	
58.0000	0.8730	1.0000	1
59.0000	0	0	
60.0000	0.0389	0	
61.0000	0.3643	0	0 **Misclassified**
62.0000	1.0000	1.0000	
63.0000	0.8174	1.0000	
64.0000	0.8875	1.0000	1
65.0000	0.7995	1.0000	
66.0000	0.5919	0	
67.0000	0.7533	1.0000	1

Table 7. Continued.

File	Membership	Defuzzified	Result
68.0000	0.7337	1.0000	
69.0000	0.8524	1.0000	
70.0000	0.8602	1.0000	1
71.0000	0.2217	0	
72.0000	1.0000	1.0000	
73.0000	0.1268	0	0 Misclassified
74.0000	0.8860	1.0000	
75.0000	0.2121	0	
76.0000	0.1684	0	
77.0000	0.6903	1.0000	0 Misclassified
78.0000	0.7680	1.0000	
79.0000	0.8735	1.0000	
80.0000	0.8013	1.0000	1
81.0000	0.1748	0	
82.0000	0.5428	0	
83.0000	0.8496	1.0000	0 Misclassified
84.0000	0.3444	0	
85.0000	0.8298	1.0000	
86.0000	0.8590	1.0000	1
87.0000	0.6879	1.0000	
88.0000	0.9082	1.0000	
89.0000	0.6653	1.0000	1
90.0000	0.1636	0	
91.0000	0.8754	1.0000	
92.0000	0.8594	1.0000	1
93.0000	0.5185	0	
94.0000	0.4932	0	
95.0000	0.7802	1.0000	0 Misclassified
96.0000	0.8684	1.0000	
97.0000	0.8788	1.0000	
98.0000	1.0000	1.0000	1
99.0000	1.0000	1.0000	
100.0000	0.8669	1.0000	1

Table 7. Continued.

File	Membership	Defuzzified	Result
1.0000	0.3986	0	
2.0000	0.2845	0	
3.0000	0.2562	0	0
4.0000	0.2786	0	
5.0000	0.3226	0	
6.0000	0	0	0
7.0000	1.0000	1.0000	
8.0000	0.5055	0	
9.0000	0.1434	0	0
10.0000	0	0	
11.0000	0	0	0
12.0000	0.0691	0	
13.0000	0.4744	0	
14.0000	0.4708	0	0
15.0000	0	0	
16.0000	0	0	
17.0000	0	0	0
18.0000	0.4623	0	0
19.0000	0	0	
20.0000	0	0	
21.0000	0.2096	0	0
22.0000	0	0	
23.0000	0	0	
24.0000	0.0516	0	0
25.0000	0.2885	0	
26.0000	0.0981	0	
27.0000	0.9336	1.0000	0
28.0000	0.2254	0	
29.0000	0.1465	0	
30.0000	0.0680	0	0
31.0000	0	0	
32.0000	0	0	
33.0000	0.0939	0	0

Table 8. Classification of the files in Set3.

File	Membership	Defuzzified	Result
34.0000	0.3917	0	
35.0000	0	0	
36.0000	0	0	0
37.0000	0.1689	0	
38.0000	0.5220	0	
39.0000	0	0	0
40.0000	0.0969	0	
41.0000	0	0	
42.0000	0	0	0
43.0000	0.4810	0	
44.0000	0.3154	0	
45.0000	0.4552	0	0
46.0000	0.3285	0	0
47.0000	0.3690	0	
48.0000	0.5593	0	
49.0000	0.3522	0	
50.0000	0.2325	0	0
51.0000	1.0000	1.0000	
52.0000	0.9052	1.0000	
53.0000	0.8115	1.0000	1
54.0000	0.8397	1.0000	
55.0000	0.8754	1.0000	
56.0000	0.0930	0	1
57.0000	0.8330	1.0000	
58.0000	1.0000	1.0000	1
59.0000	1.0000	1.0000	
60.0000	1.0000	1.0000	
61.0000	1.0000	1.0000	1
62.0000	1.0000	1.0000	
63.0000	0.6496	1.0000	
64.0000	0.5075	0	1
65.0000	0.0823	0	
66.0000	0.7810	1.0000	
67.0000	0.2356	0	0 Misclassified

Table 8. Continued.

File	Membership	Defuzzified	Result
68.0000	1.0000	1.0000	
69.0000	1.0000	1.0000	
70.0000	1.0000	1.0000	1
71.0000	1.0000	1.0000	
72.0000	1.0000	1.0000	
73.0000	1.0000	1.0000	1
74.0000	1.0000	1.0000	
75.0000	1.0000	1.0000	
76.0000	1.0000	1.0000	1
77.0000	1.0000	1.0000	
78.0000	1.0000	1.0000	
79.0000	1.0000	1.0000	1
80.0000	0.6068	1.0000	
81.0000	0.9054	1.0000	
82.0000	0.4134	0	1
83.0000	1.0000	1.0000	
84.0000	0	0	
85.0000	0.2914	0	0 **Misclassified**
86.0000	1.0000	1.0000	
87.0000	1.0000	1.0000	
88.0000	0.8786	1.0000	1
89.0000	0.9018	1.0000	
90.0000	1.0000	1.0000	
91.0000	1.0000	1.0000	1
92.0000	1.0000	1.0000	
93.0000	0.9135	1.0000	
94.0000	0.8292	1.0000	1
95.0000	0.7423	1.0000	
96.0000	1.0000	1.0000	
97.0000	0.0902	0	1
98.0000	0.2564	0	
99.0000	0	0	
100.0000	0.4387	0	0 **Misclassified**

Table 8. Continued.

Appendix B

Programs

```
function v=armod(var,M)

% This function finds the autoregressive parameter fo the signal
% and then prewhitens the signal using the prewhiten filter.
% Recursive Levinston and durbin algorithm is used to find the AR parameters

% To use the function the user should enter the signal and the AR model order
% eg armod(variable, model order)

Fs=30;                                  %sampling frequency


r=xcorr(var,'biased');                  %rx(0) is at index K
K=length(var);
rx=r(K:K+M+1);                          %rx(0),rx(1),..rx(M)


% Estimate the reflection coefficients

a(1,1)=1;
P=rx(1);

for k=0:M-1
    accum=0;
    for m=0:k
        accum=accum+a(k+1,m+1)*rx(k-m+2);
    end
    gamma(k+2)=-accum/P;
    P=P*(1-abs(gamma(k+2))^2);
    a(k+2,1)=1;
    a(k+2,k+2)=gamma(k+2);
    for m=1:k
        a(k+2,m+1)=a(k+1,m+1)+gamma(k+2)*a(k+1,k-m+2);
    end
end
parameter=a(M+1,:);

bb=[1];
aa=a(M+1,:);

v=filter(aa,bb,var);
```

```
function  freq=fundfreq(frag)

% This function called fundfreq (stands for fundamental frequency)
% finds the fundamental frequency of the desired signal.
% for the K interval of a question using autocorrelation function.
% For a periodic signal with the period p, the autocorrelation function
% attains a maximum at 0,p,2p,..
% regardless of the time origin of the signal, the period can be estimated
% by finding the location the first maximum in the autocorrelation function.


%For using this function the user should enter the file segment fundfreq(frag).

Fs = 30;                          %Sampling frequency
K=length(frag);

y = xcorr(frag);                  % finds the autocorralation function

q = diff(abs(y(K:2*K-1)));        % differentiates the variable

z = q>0;                          % z = 1 if q is greater than 0

f = diff(z);                      %finds the indices where the 2nd derivative
                                  %is -1 or +1 which indicates peaks and valleys

peak = find(f<0);                 %finds the peak indices


m =K+peak;
[i,j]=max(abs(y(m)));             %finds the maximum peak value and its index

lofreq =find(f>=0);
   if length(lofreq)==length(f)
     freq=0;
   else
     freq = Fs/peak(j);
   end
```

```
function y=croscor(var1,var2)


% This function finds the cross correlation between two variables
% The first variable is prewhitened first by calling
% armod (stands for AR modeling) program.
% The function returns maximum and minimum of the croscorrelation
% and the lag that these maximum and minimum happen.
%To use this command the user must enter the two
%variable names to be correlated.
%
% eg.   croscor(variable1,variable2)

K=min(length(var1),length(var2));



M=10;                                          % Model order
v1=armod(var1,M);

yd= xcorr(v1(20:K),var2(20:K),'biased');

[maximum lagmax]=max(real(yd));

[minimum lagmin]=min(real(yd));

y=[maximum lagmax minimum lagmin];
```

```
function feature= feature(file_name,relevant,irrelevant,control,features,offset,CR_feature)

% This function produces a feature vector for a given file
% Relevent, irrelevent, and control are vectors which contain
% the questions these features are extracted from.
%
% eg. featurev(t79,[3 5],[1 4], [6 10],feature_list)

% The above example gives the features for
% the file t79 of the 3rd and 5th question which are relevent in this
% MGQT format, the 1st and 4th question which are irrelevent
% and the 6th and 10th questions which are control

% feature_list=['10mean(frag ) ';
%               '20curve(frag )';
%               '30area(frag ) '];


feature_list = features;



% The channels are ordered as follows:
% 1:GSR, 2:HiCardio, 3:LowCardio, 4:DerLowCardio, 5:LowResp, 6:UpResp


% This is a matrix of the time delay after asking a question to start of extracting
% the feature, and finish extracting the feature for each channel.

Times=[
        2, 14;
        3, 9 ;
        3, 18;
        1, 8 ;
        2, 18;
        2, 18];

% These are preprocessing functions.
Preprocess=[  'detgsr';
              'dethic';
              'detlc ';
              'dercd ';
              'detlr ';
              'detur '];
```

```
data=zeros(6,length(file_name(:,5)));
% Standardize and detrend the channels and derive new channels

for i=1:6,
        data(i,:)=eval([Preprocess(i,:),'(file_name)']);
end


marker = file_name(:,5);     % 0 begin test and end test
                        % 0 examiner begins asking question
                        % 1 examiner finishes asking question
                        % 2 subject begins response to question
                        % 9 does not mark an event

begin = find(marker == 0);      % finds indecies where marker = 0 (question begins)
begin=begin(2:length(begin)); % elliminates the marker at the beginning of the test

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++
% This for loop creates feature vectors for each relevant quesion
%
% eg x = [mean(gsr),std(gsr),area(gsr),mean(lr),std(lr),area(lr),etc.....
% curve length,amplitude of peaks,# of peaks]
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++

feature_count=1;

for i = 1:max(find(relevant~=0)),
        question=relevant(i);

        for j=1:length(feature_list(:,1))
                        channel_number=eval(feature_list(j,1));
                        second_channel=eval(feature_list(j,2));
                        st=begin(question)+30*Times(channel_number,1);
                        fn=begin(question)+30*Times(channel_number,2);
                        st2=begin(question)-30*Times(channel_number,2);
                        fn2=begin(question)-30*Times(channel_number,1);
                        fr=feature_list(j,3:length(feature_list(1,:)));
                        frag=data(channel_number,st:fn);
                        frag2 = data(channel_number,st2:fn2);
                        if second_channel ~= 0
```

```
                        st3=begin(question)+30*Times(second_channel,1);
                        fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                    end
                    tempy=eval(fr);
                    for m = 1:length(tempy)
                        x(feature_count) = tempy(m);
                        feature_count=feature_count+1;
                    end
        end
end
%-----------------------------------------------------------------------
% Irrelevant questions

feature_count=1;

for i = 1:(max(find(irrelevant~=0))-offset)
        question=irrelevant(i);
        for j=1:length(feature_list(:,1))
                channel_number=eval(feature_list(j,1));
                second_channel=eval(feature_list(j,2));
                st=begin(question)+30*Times(channel_number,1);
                fn=begin(question)+30*Times(channel_number,2);
                st2=begin(question)-30*Times(channel_number,2);
                fn2=begin(question)-30*Times(channel_number,1);
                fr=feature_list(j,3:length(feature_list(1,:)));
                frag=data(channel_number,st:fn);
                frag2 = data(channel_number,st2:fn2);
                if second_channel ~= 0
                        st3=begin(question)+30*Times(second_channel,1);
                        fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                end
                tempy=eval(fr);
                for m = 1:length(tempy)
                        y(feature_count) = tempy(m);
                        feature_count=feature_count+1;
                end
        end
end
```

```
%-------------------------------------------------------------------
% Control questions

feature_count=1;

for i = 1:max(find(control~=0)),
        question=control(i);

        for j=1:length(feature_list(:,1))
                channel_number=eval(feature_list(j,1));
                second_channel=eval(feature_list(j,2));
                st=begin(question)+30*Times(channel_number,1);
                fn=begin(question)+30*Times(channel_number,2);
                st2=begin(question)-30*Times(channel_number,2);
                fn2=begin(question)-30*Times(channel_number,1);
                fr=feature_list(j,3:length(feature_list(1,:)));
                frag=data(channel_number,st:fn);
                frag2 = data(channel_number,st2:fn2);
                if second_channel ~= 0
                        st3=begin(question)+30*Times(second_channel,1);
                        fn3=begin(question)+30*Times(second_channel,2);
                        frag3 = data(second_channel,st3:fn3);
                end
                tempy=eval(fr);
                for m = 1:length(tempy)
                        z(feature_count) = tempy(m);
                        feature_count=feature_count+1;
                end
        end
end
%-------------------------------------------------------------------------------------------

% control & relevant


feature_count=1;

for i = 1:max(find(relevant~=0)),
                for k=1:max(find(control~=0)),
                        q(k)=abs(relevant(i)-control(k));
                end

                [a b]=min(q);
```

```
            question1=relevant(i);
            question2=control(b);

            for j=1:length(CR_feature(:,1))
                channel_number=eval(CR_feature(j,1));
                st=begin(question1)+30*Times(channel_number,1);
                fn=begin(question1)+30*Times(channel_number,2);
                st2=begin(question2)+30*Times(channel_number,1);
                fn2=begin(question2)+30*Times(channel_number,2);
                fr=CR_feature(j,3:length(CR_feature(1,:)));
                frag1=data(channel_number,st:fn);
                frag2=data(channel_number,st2:fn2);
                tempy=eval(fr);
                for m = 1:length(tempy)
                    w(feature_count) = tempy(m);
                    feature_count=feature_count+1;
                end
            end
end

feature=[x,y,z,w]';
```

```
function   isd_dif=isd(frag1,frag2)

% This is a integrated  spectral difference(isd) function that finds the cumulativespectral
% density of a control-relevant pair, then calculates the difference between the
% isd of control and the relevant for a part of a question.
% This function returns the max or min and the frequency (points)
% where this max or min happens and the area underneath  this difference.

% To use this command the user must enter the two variable names.
% The first variable is a control question fragment and the second is
% a relevant question fragment.
% eg.   isd1(variable1,variable2)

Fs = 30;
K=min(length(frag1),length(frag2));

nnp =1;
np = 2^nnp;
L = K/np;
L=2^(nextpow2(L));

M= spectrum (frag1,L);        %spectral density of the first (control) question
N= spectrum (frag2,L);        %spectral density of the second(relevant) question

pqc = cumsum(M(:,1));         %Cumulative sum of the integrated spectral density
pqr = cumsum(N(:,1));         %Cumulative sum of the integrated spectral density

clear M
clear N
hc = pqc/pqc(L/2);
hr = pqr/pqr(L/2);

CR_dif= hr' - hc';
 if (abs(max(CR_dif))>abs(min(CR_dif)))
    [CR_dif, mpoint]=max(CR_dif);
 else
    [CR_dif ,mpoint]=min(CR_dif);
 end
isd_dif=[ CR_dif  mpoint trapz(hr'-hc')];
```

```
feature_list=[   '10mean(frag)          ';
                 '10curve(frag)         ';
                 '10area(frag)          ';
                 '10med_dif(frag,8)     ';
                 '10max_min(frag)       ';
                 '10max(frag)           ';
                 '10min(frag)           ';
                 '10mdif(frag)          ';
                 '20mean(frag)          ';
                 '20curve(frag)         ';
                 '20area(frag)          ';
                 '20ampcard(frag)       ';
                 '20dampcard(frag)      ';
                 '20peaknumc(frag)      ';
                 '20med_dif(frag,5)     ';
                 '20max_min(frag)       ';
                 '20max(frag)           ';
                 '30min(frag)           ';
                 '20min(frag)           ';
                 '20mdif(frag)          ';
                 '20minampc(frag)       ';
                 '30mean(frag)          ';
                 '30curve(frag)         ';
                 '30area(frag)          ';
                 '30med_dif(frag,5)     ';
                 '30max_min(frag)       ';
                 '30max(frag)           ';
                 '30mdif(frag)          ';
                 '40mean(frag)          ';
                 '40min(frag)           ';
                 '40mdif(frag)          ';
                 '40curve(frag)         ';
                 '40area(frag)          ';
                 '40med_dif(frag,5)     ';
                 '40max_min(frag)       ';
                 '40max(frag)           ';
                 '50mean(frag)          ';
                 '50curve(frag)         ';
                 '50area(frag)          ';
                 '50ampr(frag)          ';
                 '50peaknumr(frag)      ';
                 '50ie(frag)            ';
                 '50dampr(frag)         ';
                 '50ieie(frag, frag2)   ';
                 '50med_dif(frag,8)     ';
                 '50max_min(frag)       ';
                 '50max(frag)           ';
                 '50min(frag)           ';
                 '50mdif(frag)          ';
                 '50minampr(frag)       ';
                 '60mean(frag)          ';
```

```
                    '60curve(frag)          ';
                    '60area(frag)           ';
                    '60ampr(frag)           ';
                    '60dampr(frag)          ';
                    '60peaknumr(frag)       ';
                    '60ie(frag)             ';
                    '60ieie(frag, frag2)    ';
                    '60med_dif(frag,8)      ';
                    '60max_min(frag)        ';
                    '60max(frag)            ';
                    '60min(frag)            ';
                    '60mdif(frag)           ';
                    '60minampr(frag)        ';
                    '10std(frag)            ';
                    '20std(frag)            ';
                    '30std(frag)            ';
                    '40std(frag)            ';
                    '50std(frag)            ';
                    '60std(frag)            ';
                    '20armod1(frag)         ';
                    '20cor1(frag)           ';
                    '50cor1(frag)           ';
                    '26croscor(frag,frag3)  ';
                    '26psdcoh1(frag,frag3)  '];

CR_feature=[
                    '10isd1(frag1,frag2)        ';
                    '20isd1(frag1,frag2)        '];


lf=length(feature_list(:,1));
cd \mgqt\g1
files1
for d=1:3
        if d==2
                cd \mgqt\g2
                files2
        elseif d==3
                cd \mgqt\non_dec
                filesn
        end

    for k=1:length(flist(:,1))
        file_name=[flist(k,:)];
        flength=length(file_name);
        question=['ZZ',num2str(file_name(3:flength-1)),'4'];
% creates the name of the file that holds the questions(zz*.014) .
```

```
        eval(['load ', file_name]);                % load the data & the file with the
        eval(['load ', question]);                 % question number
        file_name=file_name(1:flength-4);          %eleminates the extention(.013)
        question=question(1:flength-4) ;           % in order to use the data.
        Q=eval(question);
        l_rel=max(find(Q(2,:)~=0));                %The length of relevant questions
        l_con=max(find(Q(4,:)~=0));                %The length of control questions
        l_irr=max(find(Q(3,:)~=0));                %The length of irrelevant questions
        qover =l_con+l_rel+l_irr-10;               % finds the number of questions over 10
        offset=qover*(qover>0);
        CRlength=l_rel*6;
        size_M=(10+(qover<0)*qover)*(lf+18)+CRlength;  %total size of features

        initial=zeros(10*(18+lf)+30,1);            %Initializing M with a 10*lf zeros
        M(:,k)=initial;
        M(1:size_M,k)=feature(eval(file_name),[Q(2,:)],[Q(3,:)],[Q(4,:)],feature_list,offset,C
        R_feature);

        eval(['clear ',upper(file_name)])
        eval(['clear ',upper(question)])

    end

        save new_feat M lf flist
        clear M
end
```

```
clear
featlength=23;
load new_feat
for k=1:length(flist(:,1))
        file_name=[flist(k,:)];
        flength=length(file_name);
        question=['ZZ',num2str(file_name(3:flength-1)),'4'];
        eval(['load ',question]);              % load the file with the question numbers.
        Q=eval(question(1:flength-4));         % in order to use the data.
        l_rel=max(find(Q(2,:)~=0));            %The length of relevant questions
        l_con=max(find(Q(4,:)~=0));            %The length of control questions
        l_irr=max(find(Q(3,:)~=0));            %The length of irrelevant questions


% Averaging relevant questions
        for j=1:lf-5+featlength
                m=(j-1)*7;
                clear r
                for i=1:l_rel
                    r(i)=M((i-1)*(lf-5+featlength)+j,k);    %finds the feature values
                end                                         %for all the relevant questions.

                feat_vec(m+1,k)=mean(r);                    %returns mean value for relevant
                feat_vec(m+2,k)=mean(r);
                feat_vec(m+3,k)=max(r);
                feat_vec(m+4,k)=min(r);
                feat_vec(m+5,k)=max(r);
                feat_vec(m+6,k)=min(r);
                feat_vec(m+7,k)=max(r);
        end
  qover =l_con+l_rel+l_irr-10 ;                     %The number of questions over 10
  offset=qover*(qover>0);
  l=(l_irr-offset+l_rel)*(lf-5+featlength);         %The position of the
  cr_l=l+l_con*(lf-5+featlength);                   %first control question

%---------------------------------------

% Averaging control questions

  for j=1:lf-5+featlength
        clear c
        m=(j-1)*7;
        for i=1:l_con
                c(i)=M((i-1)*(lf-5+featlength)+j+l,k);        %finds the feature values for
```

```
        end                                          %all the control questions.

%feature values for control questions

        f(m+1,k)=feat_vec(m+1,k)-mean(c);
                if (feat_vec(m+2,k)+mean(c)==0)
                        f(m+2,k)=100;
                else
                        f(m+2,k)=2*(feat_vec(m+2,k)-
                         mean(c))/(feat_vec(m+2,k)+mean(c));   %for every feature.
                end
        f(m+3,k)=feat_vec(m+3,k)-max(c);
        f(m+4,k)=feat_vec(m+4,k)-min(c);
        f(m+5,k)=feat_vec(m+5,k)-min(c);
        f(m+6,k)=feat_vec(m+6,k)-max(c);
                if max(c)==0
                        f(m+7,k)=100;
                else
                        f(m+7,k)=feat_vec(m+7,k)/max(c);
                end
        end

%-------------------------------------------------
% feature values for control_relevant

  for j=1:6
        m=(j-1)*3;
        clear cr
        for i=1:l_rel
                cr(i)=M((i-1)*6+j+cr_l,k);
        end

        f(m+1+(lf-5+featlength)*7,k)=mean(cr);
        f(m+2+(lf-5+featlength)*7,k)=max(cr);
        f(m+3+(lf-5+featlength)*7,k)=min(cr);
  end

decep(1,k)=Q(1:1);                  % finds if file is deceptive or not
                                    % creates 1 if deceptive and 0 if not.
eval(['clear ',upper(question(1:flength-4))]);
end


save fn_dec f decep
```